Note:
Before using this information and the product it supports, be sure to read the general information under “Notices” on page 619.
Figures

1. How to code comments and continuation lines........................................................................................................... 3
2. Layout of the RPL extension (part 1 of 3)......................................................................................................................... 584
3. Layout of the RPL extension (part 2 of 3)......................................................................................................................... 585
4. Layout of the RPL extension (part 3 of 3)......................................................................................................................... 586
Tables

1. Format of WHATRCV mask........................................................................................................................343
2. LU 6.2 global macro variables set by ISTGAPPC......................................................................................521
3. Register contents upon return of control................................................................................................. 611
About this document

This manual is designed to help customers write VTAM® application programs to use the VTAM logical unit (LU) 6.2 application programming interface (API). This manual describes the format of the macroinstructions and presents each macroinstruction in alphabetical order.

This manual explains macroinstruction syntax and parameters, return codes and responses, and identifies fields set by DSECTs. z/OS Communications Server: SNA Programmer's LU 6.2 Guide also explains how to design VTAM LU 6.2 application programs.

Who should use this document

This book is for programmers (such as application or system programmers) who code VTAM application programs. This audience can include programmers who are modifying existing programs or writing new ones.

You should be familiar with LU 6.2 architecture before you write LU 6.2 programs. z/OS Communications Server: SNA Programmer's LU 6.2 Guide provides this familiarity and is, therefore, a corequisite for using the z/OS Communications Server: SNA Programmer's LU 6.2 Reference. You should also be familiar with information in z/OS Communications Server: SNA Programming.

You should also be familiar with the information in the assembler language documentation for your operating system.

How this document is organized

This document is organized into the following topics:

- Chapter 1, “LU 6.2 macroinstruction syntax and operands,” on page 1 describes all varieties of the VTAM APPCCMD macroinstruction.
- Chapter 2, “Return codes,” on page 533 describes the return codes.
- Chapter 3, “DSECTs,” on page 569 describes the LU 6.2 DSECTs.
- Chapter 4, “Summary of register usage,” on page 611 describes the summary of register usage.
- #unique_11 lists documents that provide architectural specifications for the SNA Protocol.
- Appendix B, “Accessibility,” on page 615 describes accessibility features to help users with physical disabilities.
- “Notices” on page 619 contains notices and trademarks used in this document.
- “Bibliography” on page 623 contains descriptions of the documents in the z/OS Communications Server library.

How to use this document

To use this document, you should be familiar with the basic concepts of telecommunication, SNA, and VTAM.

How to contact IBM service

For immediate assistance, visit this website: http://www.software.ibm.com/support

Most problems can be resolved at this website, where you can submit questions and problem reports electronically, and access a variety of diagnosis information.
For telephone assistance in problem diagnosis and resolution (in the United States or Puerto Rico), call the IBM Software Support Center anytime (1-800-IBM®-SERV). You will receive a return call within 8 business hours (Monday – Friday, 8:00 a.m. – 5:00 p.m., local customer time).

Outside the United States or Puerto Rico, contact your local IBM representative or your authorized IBM supplier.

If you would like to provide feedback on this publication, see “Communicating your comments to IBM” on page 651.

**Conventions and terminology that are used in this information**

Commands in this information that can be used in both TSO and z/OS UNIX environments use the following conventions:

- When describing how to use the command in a TSO environment, the command is presented in uppercase (for example, NETSTAT).
- When describing how to use the command in a z/OS UNIX environment, the command is presented in bold lowercase (for example, netstat).
- When referring to the command in a general way in text, the command is presented with an initial capital letter (for example, Netstat).

All the exit routines described in this information are *installation-wide exit routines*. The installation-wide exit routines also called installation-wide exits, exit routines, and exits throughout this information.

The TPF logon manager, although included with VTAM, is an application program; therefore, the logon manager is documented separately from VTAM.

Samples used in this information might not be updated for each release. Evaluate a sample carefully before applying it to your system.

**Note:** In this information, you might see the following Shared Memory Communications over Remote Direct Memory Access (SMC-R) terminology:

- **RoCE Express**, which is a generic term representing IBM 10 GbE RoCE Express, IBM 10 GbE RoCE Express2, and IBM 25 GbE RoCE Express2 feature capabilities. When this term is used in this information, the processing being described applies to both features. If processing is applicable to only one feature, the full terminology, for instance, IBM 10 GbE RoCE Express will be used.

- **RoCE Express2**, which is a generic term representing an IBM RoCE Express2® feature that might operate in either 10 GbE or 25 GbE link speed. When this term is used in this information, the processing being described applies to either link speed. If processing is applicable to only one link speed, the full terminology, for instance, IBM 25 GbE RoCE Express2 will be used.

- **RDMA network interface card (RNIC)**, which is used to refer to the IBM® 10 GbE RoCE Express, IBM® 10 GbE RoCE Express2, or IBM 25 GbE RoCE Express2 feature.

- **Shared RoCE environment**, which means that the "RoCE Express" feature can be used concurrently, or shared, by multiple operating system instances. The feature is considered to operate in a shared RoCE environment even if you use it with a single operating system instance.

**Clarification of notes**

Information traditionally qualified as Notes is further qualified as follows:

**Attention**

- Indicate the possibility of damage

**Guideline**

- Customary way to perform a procedure

**Note**

- Supplemental detail
Rule
Something you must do; limitations on your actions

Restriction
Indicates certain conditions are not supported; limitations on a product or facility

Requirement
Dependencies, prerequisites

Result
Indicates the outcome

Tip
Offers shortcuts or alternative ways of performing an action; a hint

Prerequisite and related information

z/OS Communications Server function is described in the z/OS Communications Server library. Descriptions of those documents are listed in “Bibliography” on page 623, in the back of this document.

Required information

Before using this product, you should be familiar with TCP/IP, VTAM, MVS™, and UNIX System Services.

Softcopy information

Softcopy publications are available in the following collection.

<table>
<thead>
<tr>
<th>Titles</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Z Redbooks</td>
<td>The IBM Z™ subject areas range from e-business application development and enablement to hardware, networking, Linux, solutions, security, parallel sysplex, and many others. For more information about the Redbooks® publications, see <a href="http://www.redbooks.ibm.com/">http://www.redbooks.ibm.com/</a> and <a href="http://www.ibm.com/systems/z/os/zos/zfavorites/">http://www.ibm.com/systems/z/os/zos/zfavorites/</a>.</td>
</tr>
</tbody>
</table>

Other documents

This information explains how z/OS references information in other documents.

When possible, this information uses cross-document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see z/OS Information Roadmap (SA23-2299). The Roadmap describes what level of documents are supplied with each release of z/OS Communications Server, and also describes each z/OS publication.

To find the complete z/OS library, visit the z/OS library in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/SSLTBW/welcome).

Relevant RFCs are listed in an appendix of the IP documents. Architectural specifications for the SNA protocol are listed in an appendix of the SNA documents.

The following table lists documents that might be helpful to readers.

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
</tr>
</thead>
</table>
### Redbooks publications

The following Redbooks publications might help you as you implement z/OS Communications Server.

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM z/OS Communications Server TCP/IP Implementation, Volume 1: Base Functions, Connectivity, and Routing</td>
<td>SG24-8096</td>
</tr>
<tr>
<td>IBM z/OS Communications Server TCP/IP Implementation, Volume 2: Standard Applications</td>
<td>SG24-8097</td>
</tr>
<tr>
<td>IBM z/OS Communications Server TCP/IP Implementation, Volume 3: High Availability, Scalability, and Performance</td>
<td>SG24-8098</td>
</tr>
<tr>
<td>IBM z/OS Communications Server TCP/IP Implementation, Volume 4: Security and Policy-Based Networking</td>
<td>SG24-8099</td>
</tr>
</tbody>
</table>
Where to find related information on the Internet

**z/OS**

This site provides information about z/OS Communications Server release availability, migration information, downloads, and links to information about z/OS technology

http://www.ibm.com/systems/z/os/zos/

**z/OS Internet Library**

Use this site to view and download z/OS Communications Server documentation


**IBM Communications Server product**

The primary home page for information about z/OS Communications Server


**z/OS Communications Server product**

The page contains z/OS Communications Server product introduction


**IBM Communications Server product support**

Use this site to submit and track problems and search the z/OS Communications Server knowledge base for Technotes, FAQs, white papers, and other z/OS Communications Server information

http://www.software.ibm.com/support

**IBM Communications Server performance information**

This site contains links to the most recent Communications Server performance reports

http://www.ibm.com/support/docview.wss?uid=swg27005524

**IBM Systems Center publications**

Use this site to view and order Redbooks publications, Redpapers, and Technotes

http://www.redbooks.ibm.com/
IBM Systems Center flashes
Search the Technical Sales Library for Techdocs (including Flashes, presentations, Technotes, FAQs, white papers, Customer Support Plans, and Skills Transfer information)
http://www.ibm.com/support/techdocs/atsmastr.nsf

Tivoli® NetView® for z/OS
Use this site to view and download product documentation about Tivoli NetView for z/OS
http://www.ibm.com/support/knowledgecenter/SSZJDU/welcome

RFCs
Search for and view Request for Comments documents in this section of the Internet Engineering Task Force website, with links to the RFC repository and the IETF Working Groups web page
http://www.ietf.org/rfc.html

Internet drafts
View Internet-Drafts, which are working documents of the Internet Engineering Task Force (IETF) and other groups, in this section of the Internet Engineering Task Force website
http://www.ietf.org/ID.html

Information about web addresses can also be found in information APAR II11334.

Note: Any pointers in this publication to websites are provided for convenience only and do not serve as an endorsement of these websites.

DNS websites
For more information about DNS, see the following USENET news groups and mailing addresses:

USENET news groups
comp.protocols.dns.bind

BIND mailing lists
https://lists.isc.org/mailman/listinfo

BIND Users
- Subscribe by sending mail to bind-users-request@isc.org.
- Submit questions or answers to this forum by sending mail to bind-users@isc.org.

BIND 9 Users (This list might not be maintained indefinitely.)
- Subscribe by sending mail to bind9-users-request@isc.org.
- Submit questions or answers to this forum by sending mail to bind9-users@isc.org.

The z/OS Basic Skills Information Center
The z/OS Basic Skills Information Center is a web-based information resource intended to help users learn the basic concepts of z/OS, the operating system that runs most of the IBM mainframe computers in use today. The Information Center is designed to introduce a new generation of Information Technology professionals to basic concepts and help them prepare for a career as a z/OS professional, such as a z/OS systems programmer.

Specifically, the z/OS Basic Skills Information Center is intended to achieve the following objectives:
- Provide basic education and information about z/OS without charge
- Shorten the time it takes for people to become productive on the mainframe
- Make it easier for new people to learn z/OS
To access the z/OS Basic Skills Information Center, open your web browser to the following website, which is available to all users (no login required): https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zbasics/homepage.html?cp=zosbasics
Summary of changes

This document contains terminology, maintenance, and editorial changes, including changes to improve consistency and retrievability. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Changes made in z/OS Communications Server Version 2 Release 3

This information contains no technical change for this release.

Changes made in z/OS Communications Server Version 2 Release 2

This information contains no technical change for this release.

z/OS Version 2 Release 1 summary of changes

See the Version 2 Release 1 (V2R1) versions of the following publications for all enhancements related to z/OS V2R1:

- z/OS Migration
- z/OS Planning for Installation
- z/OS Summary of Message and Interface Changes
- z/OS Introduction and Release Guide
Chapter 1. LU 6.2 macroinstruction syntax and operands

About this chapter

This chapter describes all varieties of the VTAM APPCCMD macroinstruction. Separate descriptions are included for each CONTROL and QUALIFY combination of the macroinstruction. This chapter also includes the ISTGAPPC and ISTRPL6 macroinstructions. Macroinstruction descriptions are arranged alphabetically.

The macroinstructions are coded as assembler instructions, using name, operation, and operand fields. Refer to the IBM High Level Assembler Language Reference for MVS and VM for coding guidelines.

How macroinstructions are described

Each macroinstruction description contains:

• The purpose of the macroinstruction
• General comments about its use
• Reference to tutorial chapters detailing its use
• The syntax of the macroinstruction and all parameters
• Input parameter descriptions
• RPL and RPL extension fields set by the macroinstruction
• Return and reason codes that can be returned for the macroinstruction

Syntax descriptions

The syntax for the macroinstruction is described using the following format:

```
name    APPCCMD    Required and Optional Operands
```

**Note:** If you are not familiar with this type of syntax diagram, see “How to read the syntax diagrams” on page 4.

**Name field**

The *name* field provides a label for the macroinstruction. If you use a name, it must begin in column one of the macroinstruction, be followed by one or more blanks, and contain 1 to 8 characters in the following format:

<table>
<thead>
<tr>
<th>Character</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>First character</td>
<td>Alphabetical (A-Z) or the national characters @, #, or $</td>
</tr>
<tr>
<td>Second to eighth character</td>
<td>Alphabetical (A-Z), numerical (0-9), or the national characters @, #, or $</td>
</tr>
</tbody>
</table>

**The macroinstruction field**

The APPCCMD identifier is always coded exactly as shown. It begins in column ten and must be preceded and followed by one or more blanks.
Operands can occupy columns 16-71. You must place one or more blanks after the last operand on a line. If the operands require more than one line, you must place a nonblank continuation character in column 72. (See “Coding continued lines” on page 3.)

Operands consist of a fixed character string (the operand keyword) followed by an equal sign (=) and one or more operand values. If the value is specified as name, it must follow the rules for a symbolic name detailed in *IBM High Level Assembler Language Reference for MVS and VM*.

Operands do not have to be coded in the order shown by the syntax diagram. For example, a macroinstruction having the operands AREALEN=data_length and AREA=data_area_address could be coded in either of two ways:

```
AREALEN=132, AREA=WORK
AREA=WORK, AREALEN=132
```

Keyword operands must be separated by commas. If you choose to omit a keyword operand, also omit the comma that would have been included with it.

When more than one value can be coded after the keyword, a parenthesis is required. The OPTCD keyword can specify the manner of processing (asynchronous or synchronous) and control use of the buffer list option. For example, code the following information to specify synchronous and buffer list options: OPTCD=(SYN,BUFFLST)

**Operand descriptions**

Each operand description begins with an explanation of the operand function. If the operand allows more than one fixed value that can be supplied with it, the operand description explains the effect that each value has on the action performed by the macroinstruction.

Most operands are coded by these general rules. If the format varies from these rules, the format is included with the description.

- When a quantity is indicated (for example, RECLEN=data_length), you can specify the value with:
  - Decimal integers.
  - An expression that is equal to such a value (for example, RECLEN=TEN, where TEN EQU 5*2).
  - The number of the register (enclosed by parentheses) to contain the quantity when the macroinstruction is executed. When specifying a quantity, Register notation is restricted to registers 2-12.
- When an address is indicated (for example, AREA=data_area_address), you can use any expression valid for an RX-type assembler instruction (for example, an LA instruction). Registers 1-12 can be specified for any operand that designates the address of an RPL. Register notation for all other address operands is restricted to registers 2-12.

For more information on operand formats, refer to the assembler language publication appropriate to your operating system.

**Completion information**

All executable macroinstructions pass return codes in registers, and most indicate status information in control block fields when they are posted complete. This status information is described at the end of the macroinstruction description. Refer to *z/OS Communications Server: SNA Programmer’s LU 6.2 Guide* for information regarding the sequence of error checking.
Coding default values

The default values apply only to declarative macroinstructions ACB, EXLST, NIB, ISTRPL6, and RPL. All other macroinstructions (including APPCCMDs) use values specified in the macroinstruction itself or currently defined in the referenced control blocks. APPCCMDs do not have defaults; the defaults are in the underlying RPL and RPL extension.

Coding comments

Comments can contain any characters valid in the assembler language. You can write comments after the operand field, but they must meet the following criteria:

- Comments must begin in column 17 or beyond.
- Comments must be separated from the last operand in the field by one or more blanks.
- Comments must not extend into the continuation column, column 72.

Comments can be continued on more than one line by placing an asterisk (*) in column one as shown in Figure 1 on page 3.

```
LABEL1  OP1  OPERAND1,OPERAND1,OPERAND3,OPERAND4,OPERAND5  THIS IS ONE WAY
        ,
LABEL2  OP2  OPERAND1,OPERAND2, AND THIS IS ANOTHER WAY
        *
```

*Figure 1. How to code comments and continuation lines*

**Note:** A macroinstruction that has no operands cannot have comments on APPCCMD identifier line.

An entire line can be used for a comment; place an asterisk in the first column of the line. If you need several entire lines for comments, place an asterisk in the first column of each line and leave column 72 blank.

In this manual, the comments field is not shown in the macroinstruction syntax diagram.

Coding continued lines

**Procedure**

Code VTAM macroinstructions in columns 1-71 of a line. Macroinstructions that exceed 71 columns can be continued on additional lines as shown in Figure 1 on page 3. (Continuation characters are omitted from other examples in this manual.) When you need to continue on another line, the following steps apply:

1. Code the macroinstruction one of two ways:
   - Through column 71
   - Through any completed operand, stopping after the comma that separates the operand from those that follow

2. Enter a nonblank continuation character in column 72.
The continuation character is not considered to be part of the macroinstruction.

3. Continue operands beginning in column 16 of the next line.
   Columns 1-15 must be blank. A continuation line that begins in column 17 or later is ignored. A comment line cannot follow a continuation line.

4. If you must continue on another line, proceed with Step “1” on page 3.
5. Macroinstructions can be coded on as many lines as needed.
6. Comments can appear on every line of a continued macroinstruction.
7. Columns 73-80 can be used to code identification characters, macroinstruction sequence characters, or both.

How to read the syntax diagrams

Purpose
This section describes how to read the syntax diagrams used in this book.

- Read the diagrams from left-to-right, top-to-bottom, following the main path line. Each diagram begins on the left with double arrowheads (►►) and ends on the right with two arrowheads facing each other (◄◄).

Syntax Diagram

- If a diagram is longer than one line, the first line ends with a single arrowhead (►) and the second line begins with a single arrowhead.

First Line

Second Line

- Required operands and values appear on the main path line.

REQUIRED_OPERAND

You must code required operands and values.

If your choices are greater than one, the choices are stacked vertically in alphanumeric order.

REQUIRED_OPERAND_OR_VALUE_1

REQUIRED_OPERAND_OR_VALUE_2

- Optional operands and values appear below the main path line.

OPERAND

You can choose not to code optional operands and values.

If your choices are more than one, they are stacked vertically in alphanumeric order below the main path line.

OPERAND_OR_VALUE_1

OPERAND_OR_VALUE_2

- An arrow returning to the left above an operand or value on the main path line means that the operand or value can be repeated. The comma means that each operand or value must be separated from the next by a comma.
An arrow returning to the left above a group of operands or values means that more than one can be selected, or a single one can be repeated.

A word in all uppercase is an operand or value you must spell exactly as shown. In this example, you must code `OPERAND`.

**Note:** VTAM commands are not case-sensitive. You can code them in uppercase or lowercase.

If an operand or value can be abbreviated, the abbreviation is discussed in the text associated with the syntax diagram.

If a diagram shows a character that is not alphanumeric (parentheses, periods, commas, and equal signs), you must code the character as part of the syntax. In this example, you must code `OPERAND=(001,0.001)`.

If a diagram shows a blank space, you must code the blank space as part of the syntax. In this example, you must code `OPERAND=(001 FIXED)`.

Default operands and values appear above the main path line. VTAM uses the default if you omit the operand entirely.

A word appearing in all lowercase italics is a *variable*. Where you see a variable in the syntax, you must replace it with one of its allowable names or values, as defined in the text.

References to syntax notes appear as numbers enclosed in parentheses above the line. Do not code the parentheses or the number.

Notes:

1. An example of a syntax note.

Some diagrams contain syntax fragments; these serve to break up diagrams that are too long, too complex, or too repetitious. Syntax fragment names appear in mixed case and are shown in the diagram and in the heading of the fragment. The fragment is placed below the main diagram.
APPCCMD CONTROL=ALLOC, QUALIFY=ALLOCD

**Purpose**

This macroinstruction allocates resources for a conversation and assigns a contention winner or contention loser session to the conversation. If a session is not available and session limits allow, VTAM activates a session for the conversation, if possible.

**Usage**

QUALIFY=ALLOCD is used when an application program allocates a conversation and wants VTAM to queue the request if the request cannot be met immediately. This macroinstruction corresponds to the ALLOCATE RETURN_CONTROL (WHEN_SESSION_ALLOCATED) verb in the LU 6.2 architecture. This macroinstruction completes when VTAM assigns a session to the conversation or when an error occurs that prevents VTAM from assigning a session.

VTAM finds a session for the conversation as follows:

- If a session is free, VTAM assigns it to the conversation.
- If no free sessions exist and session limits allow, VTAM establishes a session and assigns it to the conversation.
- If a new session cannot be established, VTAM queues the request until a session becomes available or until the session limits are changed to allow the establishment of a new session.

When a conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for details on allocating a conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is suspended for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```plaintext
APPCCMD CONTROL=ALLOC, QUALIFY=ALLOCD

, RPL = rpl_address_field
```

1. `rpl_address_register`
AAREA = rpl_extension_address_field (rpl_extension_address_register)

ACB = acb_address_field (acb_address_register)

AREA = fmh-5_and_opt_pip_gds_var_add_field (fmh-5_and_opt_pip_gds_var_add_reg)

BRANCH = NO

CONMODE = BUFFCA

CONXMOD = CA

ECB = INTERNAL

EXIT = exit_routine_address_field (exit_routine_address_register)

LOGMODE = 8-byte_logon_mode_name

LUAffIN = APPL

LUNAME = 8-byte_lu_name

NAMEUSE = APNAME

LU 6.2 macroinstruction syntax and operands
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:
**AAREA**=

AAREA=**(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=

ACB=**(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**=

AREA=**(fmh-5_and_opt._pip_gds_var._add._reg.)**

Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 577 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**
  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**
  
  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- **CONMODE=BUFFCA**
  
  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

- **CONMODE=CS**
  
  Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

- **CONMODE=LLCA**
  
  Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.
CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN
Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

LUAFFIN=APPL
The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL
VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU,
the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**LUNAME=8-byte_lu_name**
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**
The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**
The application identifies itself to the partner LU by a generic resource name.
The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.
RECLLEN=fmh-5_and_opt._pip_gds_var._len.
RECLLEN=(fmh-5_and_opt._pip_gds_var._len._reg.)

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4-bytes_of_user_data
USERFLD=(user_data_register)

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

AVFA

The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B’1’)

The partner LU accepts the already-verified indicator.

NO (B’0’)

The partner LU does not accept the already-verified indicator.
CGID
Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

CONSTATE
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.
This field can have the following values for half-duplex conversations:
X'00'
  RESET
X'01'
  SEND
X'08'
  END_CONVERSATION
This field can have the following values for full-duplex conversations:
X'00'
  RESET
X'80'
  FDX_RESET
X'81'
  SEND/RECEIVE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

CONVSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.
**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**LUAFFIN**
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is returned at completion only if a requested value is specified when the macroinstruction is issued.

**NONE (B'00')**
GR affinity is not applicable or is unknown.

**NOTAPPL (B'01')**
GR affinity is not application-owned.

**APPL (B'10')**
GR affinity is application-owned.

**PRSISTVP**
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**
The partner LU accepts persistent-verification indicators.

**NO (B'0')**
The partner LU does not accept persistent-verification indicators.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**

The session was established using session-level LU-LU verification.

**NO (B'0')**

The session was not established using session-level LU-LU verification.

VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversations state is SEND after successful processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_VALID_FOR_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL_DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>RCPR</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMHSUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT.OR.DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE.</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=ALLOC, QUALIFY=CONVGRP

Purpose

This macroinstruction allocates resources for a conversation and assigns to the conversation a session with a specified conversation group identifier. If the specific session is not available and session limits allow, VTAM queues the request until the session becomes available. If the specific session does not exist, VTAM fails the allocation request.

Usage

QUALIFY=CONVGRP is used to allocate a conversation over a specific session that already exists. It provides the ability to serially allocate a related group of conversations on a particular session. This macroinstruction corresponds to the ALLOCATE RETURN_CONTROL (WHEN_CONVERSATION_GROUP_ALLOCATED) verb in the LU 6.2 architecture. This macroinstruction completes when:

- VTAM assigns the specified session to the conversation.
- The specific session is deactivated.
- An error occurs that prevents VTAM from assigning the session to the conversation.

To indicate the session to be used, the application program specifies the conversation group identifier for the session on the CGID keyword. This value was returned to the application program by the CGID returned field for a previous APPCCMD CONTROL=ALLOC, CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction.

VTAM finds the session for the conversation as follows:

- If the specified session is available, VTAM assigns it to the conversation.
- If the specified session exists but is not available, VTAM queues the request until the session becomes available.
- If the specified session is deactivated while the request is queued, the queued request will be rejected.

As with other ALLOC macroinstructions, when the conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for details on allocating a conversation.

Context

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax

```
APPCCMD CONTROL=ALLOC, QUALIFY=CONVGRP
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** *rpl_extension_address_field*

*AAREA=* *(rpl_extension_address_register)*

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.
**ACB=acb_address_field**  
**ACB=(acb_address_register)**  
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=fmh-5_and_opt._pip_gds_var._add._field**  
**AREA=(fmh-5_and_opt._pip_gds_var._add._reg.)**  
Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 577 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

**BRANCH**  
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**  
  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CGID=32-bit_conversation_group_id_field**  
**CGID=(32-bit_conversation_group_id_register)**  
Specifies the 32-bit conversation group ID. This value can be obtained from a previous APPCCMD CONTROL=ALLOC, CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction. If the CGID operand is not specified, VTAM uses the conversation group ID that is already in the RPLCGID field on the RPL extension.

The conversation group ID identifies a specific session between two specific LUs. It provides a means by which a VTAM LU 6.2 application program and its partner LU can share serially the same session.

**CONMODE**  
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- **CONMODE=BUFFCA**  
  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

- **CONMODE=CS**  
  Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

- **CONMODE=LLCA**  
  Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data.
format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.
NAMEUSE=APNAME
The application identifies itself to the partner LU by its application network name.

NAMEUSE=GNAME
The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=fmh-5_and_opt._pip_gds_var_len.
RECLEN=(fmh-5_and_opt._pip_gds_var_len._reg.)
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.
VTRINA=vector_address_field
VTRINA=(vector_address_register)
   Specifies the address of the data area where VTAM places vector list information for the application.
   This parameter is ignored if one of the following items is true:
   • VTRINA=0
   • The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
   • The value for VTRINL is not specified.
   This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
   Specifies the length of the data area where VTAM places vector list information for the application.
   This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields.

AVFA
   The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.
   YES (B'1')
      The partner LU accepts the already-verified indicator.
   NO (B'0')
      The partner LU does not accept the already-verified indicator.

CONSTATE
   The field in the RPL extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.
   This field can have the following values for half-duplex conversations:
   X'00'
      RESET
   X'01'
      SEND
   X'08'
      END_CONVERSATION
   This field can have the following values for full-duplex conversations:
   X'00'
      RESET
   X'80'
      FDX_RESET
   X'81'
      SEND/RECEIVE

CONVID
   Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.
   Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.
CONVSECP

The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')

The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')

The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL

Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')

No data is to be encrypted.

SELECTIVE (B'01')

The application program specifies the data that is to be encrypted.

REQUIRED (B'11')

All data is to be encrypted.

FDB2

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')

No FMH-5s are waiting to be received by the application program.

LOGMODE

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.
LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is returned at completion only if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

LUNAME
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NETID
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

PRISISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

SESSID
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.
**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**
The session was established using session-level LU-LU verification.

**NO (B'0')**
The session was not established using session-level LU-LU verification.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

**State changes**
These changes are applicable when RCPRI indicates OK.
For half-duplex conversation, the conversation state is SEND after successful processing.
For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.
See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_VALID_FOR_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL_DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMHS5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=ALLOC, QUALIFY=CONWIN

Purpose

This macroinstruction allocates resources for a conversation and, if session limits allow, assigns a contention-winner session to the conversation. If a contention-winner session is not available, VTAM queues the request for later completion.

Usage

QUALIFY=CONWIN is used when an application program allocates a conversation and wants VTAM to queue the request if no contention-winner session can be assigned. This macroinstruction corresponds to the ALLOCATE_RETURN_CONTROL (WHEN_CONTENTION_WINNER_ALLOCATED) verb in the LU 6.2 architecture. This macroinstruction completes when VTAM assigns a contention-winner session or an error occurs that prevents VTAM from assigning a session.

VTAM finds a session for the conversation as follows:

- If a contention-winner session is currently available, VTAM assigns it to the conversation.
- If no contention-winner session is available and session limits allow, VTAM establishes a new contention-winner session and assigns it to the conversation.
- If a new contention-winner session cannot be established, VTAM queues the request until a contention-winner session is available or session limits are changed to allow a new contention-winner session to be activated.
For this macroinstruction to complete successfully, the session limits must define at least one contention-winner session.

If contention-winner sessions are deactivated under normal conditions and an APPCCMD CONTROL=ALLOC, QUALIFY=CONWIN request is queued, VTAM activates another contention-winner session to meet the queued request.

The application program can specify how expedited data is to be received.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for details on allocating a conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```
name APPCCMD -- CONTROL = ALLOC -- QUALIFY = CONWIN

RPL = rpl_address_field
    ( rpl_address_register )

AAREA = rpl_extension_address_field
    ( rpl_extension_address_register )

ACB = acb_address_field
    ( acb_address_register )

AREA = fmh-5_and_opt_pip_gds_var_add_field
    ( fmh-5_and_opt_pip_gds_var_add_reg )

BRANCH = NO
    YES

CONMODE = BUFFCA
    CS
    LLCA
```
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters
The following information shows descriptions of the input parameters:

**AAREA**=

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**=

**AREA=(fmh-5_and_opt._pip_gds_var._add._reg.)**

Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 577 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.
EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN
Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

LUAFFIN=APPL
  The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL
  VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

LUNAME=8-byte_lu_name
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NAMEUSE
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

NAMEUSE=APNAME
  The application identifies itself to the partner LU by its application network name.

NAMEUSE=GNAME
  The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained,
the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

- **OPTCD=SYN**
  Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

- **OPTCD=ASY**
  Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

- **OPTCD=KEEPSRB**
  Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

- **OPTCD=NKEEPSRB**
  Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=**

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

**RPL=**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

- **RTSRTRN=BOTH**
  Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

- **RTSRTRN=EXPD**
  Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.
USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
   Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
   Specifies the address of the data area where VTAM places vector list information for the application.
   This parameter is ignored if one of the following items is true:
   • VTRINA=0
   • The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
   • The value for VTRINL is not specified.
   This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
   Specifies the length of the data area where VTAM places vector list information for the application.
   This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

AVFA
   The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.
   YES (B’1’)
      The partner LU accepts the already-verified indicator.
   NO (B’0’)
      The partner LU does not accept the already-verified indicator.

CGID
   Specifies the 32-bit conversation group identifier.
   It is labeled RPL6CGID in the RPL extension.

CONSTATE
   The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.
   This field can have the following values for half-duplex conversations:
   X'00'
      RESET
   X'01'
      SEND
   X'08'
      END_CONVERSATION

In addition to the half-duplex conversation states, this field can contain the following full-duplex conversation states:
**CONVID**
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

**CONVSECP**
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

**YES (B'1')**
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

**NO (B'0')**
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

**NONE (B'00')**
No data is to be encrypted.

**SELECTIVE (B'01')**
The application program specifies the data that is to be encrypted.

**REQUIRED (B'11')**
All data is to be encrypted.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.
LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

**NONE (B'00')**
- GR affinity is not applicable or is unknown.

**NOTAPPL (B'01')**
- GR affinity is not application-owned.

**APPL (B'10')**
- GR affinity is application-owned.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**
- The partner LU accepts persistent-verification indicators.

**NO (B'0')**
- The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

SESSID
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**
- The session was established using session-level LU-LU verification.

**NO (B'0')**
- The session was not established using session-level LU-LU verification.
Vectors returned

VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner’s application capabilities vector (X'1A')

State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is SEND state after successful processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See the Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_VALID_FOR_FULL DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL_DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMH5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=ALLOC, QUALIFY=IMMED

#### Purpose
This macroinstruction allocates resources for a conversation and if session limits allow, assigns an active contention-winner session to it. If no session is available, the allocation request fails.

#### Usage
QUALIFY=IMMED is used to allocate a conversation when the application program needs an immediate response from VTAM. This macroinstruction completes successfully only when an active contention-winner session is available to be assigned to the conversation. If the request cannot be met immediately, VTAM does not queue it. VTAM neither tries to activate a new session nor bids on a contention-loser session. APPCCMD CONTROL=ALLOC, QUALIFY=IMMED corresponds to the ALLOCATE RETURN_CONTROL(IMMEDIATE) verb in the LU 6.2 architecture.

When a conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program must associate a conversation with a particular transaction by using the conversation identifier.

The application program can specify how expedited data is to be received.
For details on allocating a conversation, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```
APPCCMD
   +----+----+
   | name |
   +----+----+
      IMMED 1

   +----+----+
   | RPL  |
   +----+----+
      rpl_address_field
           (--- rpl_address_register ---)

   +----+----+
   | AAREA|
   +----+----+
      rpl_extension_address_field
           (--- rpl_extension_address_register ---)

   +----+----+
   | ACB  |
   +----+----+
      acb_address_field
           (--- acb_address_register ---)

   +----+----+
   | AREA |
   +----+----+
      fmh-5_and_opt_pip_gds_var_add_field
           (--- fmh-5_and_opt_pip_gds_var_add_reg. ---)

   +----+----+
   | BRANCH |
   +----+----+
      NO 3
      YES

   +----+----+
   | CONMODE |
   +----+----+
      BUFFCA 1
      CS
      LLCA

   +----+----+
   | CONXMOD |
   +----+----+
      CA 1
      CS
```
Notes:
Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

ECB is meaningful only for asynchronous operations.

EXIT is meaningful only for asynchronous operations.

You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
  Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
  Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=fmh-5_and_opt_pip_gds_var_addr_field
AREA=(fmh-5_and_opt_pip_gds_var_addr_reg.)
  Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 577 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

BRANCH
  Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

  BRANCH=NO
    Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

  BRANCH=YES
    Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
  Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.
CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Reference.
Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

**LUNAME=8-byte_lu_name**
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NETID=8-byte_network_identifier**
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**RECLEN=fmh-5_and_opt._pip_gds_var._len.**
**RECLEN=(fmh-5_and_opt._pip_gds_var._len._reg.)**
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.
RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:
- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.
This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B’1’)
The partner LU accepts the already-verified indicator.

NO (B’0’)
The partner LU does not accept the already-verified indicator.

CGID
Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

CONSTATE
The field in the RPL extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.
This field can have the following values for half-duplex conversations:
- X’00’
  
  RESET

- X’01’
  
  SEND

- X’08’
  
  END_CONVERSATION
This field can have the following values for full-duplex conversations:

- **X'00'**
  - RESET

- **X'80'**
  - FDX_RESET

- **X'81'**
  - SEND/RECEIVE

**CONVID**

Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

**CONVSECP**

The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

- **YES (B'1')**
  - The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

- **NO (B'0')**
  - The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**

Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

- **NONE (B'00')**
  - No data is to be encrypted.

- **SELECTIVE (B'01')**
  - The application program specifies the data that is to be encrypted.

- **REQUIRED (B'11')**
  - All data is to be encrypted.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  - One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**
  - No FMH-5s are waiting to be received by the application program.
**PRSISTVP**
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

- **YES (B'1')**
  The partner LU accepts persistent-verification indicators.
- **NO (B'0')**
  The partner LU does not accept persistent-verification indicators.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

- **YES (B'1')**
  The session was established using session-level LU-LU verification.
- **NO (B'0')**
  The session was not established using session-level LU-LU verification.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

**State changes**
These changes are applicable when RCPRI indicates OK.
For half-duplex conversations, the conversation state is SEND after successful processing.
For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.
See the Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.
## Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL NOT_SUPPORTED_ BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL NOT_VALID_ FOR_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU PAIR NOT SUPPORTING_ FULL_DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE MUST BE RESTORED BEFORE USING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMHS_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED FOR NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0058'</td>
<td>X'0000'</td>
<td>UNSUCCESSFUL,—SESSION NOT AVAILABLE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE FOR YOUR ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION NOT APPCC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME RETURNED.Differs_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=ALLOC, QUALIFY=WHENFREE

#### Purpose
This macroinstruction allocates resources for a conversation and if session limits allow, assigns a session to the conversation. If a session is not available and one cannot be activated, VTAM returns control to the application program.

#### Usage
QUALIFY=WHENFREE is used when an application program allocates a conversation and wants VTAM to search for a session that satisfies the ALLOCATE request. This macroinstruction corresponds to the ALLOCATE RETURN_CONTROL (WHEN_SESSION_FREE) verb in the LU 6.2 architecture. This macroinstruction completes when VTAM assigns a session to the conversation or when VTAM cannot assign a session and returns control to the application program with a return code of X'0004', X'0001'.

VTAM finds a session for the conversation as follows:

1. If a session is available, VTAM assigns it to the conversation.
2. If no available sessions exist and session limits allow, VTAM establishes a session and assigns it to the conversation.
3. If a session cannot be established and session activation requests are pending, VTAM queues the ALLOCATE request until the request is satisfied or until all pending session activation requests are used. If the pending session activation requests are used before the ALLOCATE request is satisfied, VTAM fails the ALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001'.
4. If a session cannot be established and no session activation request is pending that might satisfy the ALLOCATE request, VTAM fails the ALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001' and returns control to the application program.

When a conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for details on allocating a conversation.

#### Context
This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

---

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
Syntax

```
name  APPCCMD  CONTROL  =  ALLOC  ,  QUALIFY  =

WHENFREE  1

RPL  =  rpl_address_field
     ( rpl_address_register )

AAREA  =  rpl_extension_address_field
       ( rpl_extension_address_register )

ACB  =  acb_address_field
     ( acb_address_register )

AREA  =  fmh-5_and_opt_pip_gds_var_add_field
       ( fmh-5_and_opt_pip_gds_var_add_reg )

BRANCH  =  NO  YES

CONMODE  =  BUFFCA  CS  LLCA

CONXMOD  =  CA  CS

ECB  =  INTERNAL
     ecb_address_field
     ( ecb_address_register )

EXIT  =  exit_routine_address_field
       ( exit_routine_address_register )

LOGMODE  =  8-byte_logon_mode_name
```
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

ECB is meaningful only for asynchronous operations.

EXIT is meaningful only for asynchronous operations.

You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPSRB is meaningful only for synchronous operations.

Input parameters

AAREA=\textit{rpl\_extension\_address\_field}

\textit{AAREA=(rpl\_extension\_address\_register)}

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\textit{acb\_address\_field}

\textit{ACB=(acb\_address\_register)}

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=\textit{fmh\textunderscore 5\_and\_opt\_pip\_gds\_var\_add\_field}

\textit{AREA=(fmh\textunderscore 5\_and\_opt\_pip\_gds\_var\_add\_reg.)}

Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 577 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE,
 QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**

*ECB=(ecb_address_register)*

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE=8-byte_logon_mode_name**

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.
**LUAFFIN=APPL**  
The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**  
VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**LUNAME=8-byte_lu_name**  
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**  
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**  
The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**  
The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**  
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**  
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**  
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**  
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=\text{fmh-5\_and\_opt\_pip\_gds\_var\_len.}**
**RECLEN=(\text{fmh-5\_and\_opt\_pip\_gds\_var\_len\_reg.})**
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

**RPL=\text{rpl\_address\_field}**
**RPL=(\text{rpl\_address\_register})**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

**RTSRTRN=EXPD**
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

**USERFLD=\text{4\_bytes\_of\_user\_data}**
**USERFLD=(\text{user\_data\_register})**
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**VTRINA=\text{vector\_address\_field}**
**VTRINA=(\text{vector\_address\_register})**
Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

**VTRINL=\text{vector\_length\_field}**
**VTRINL=(\text{vector\_length\_register})**
Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.
RPL and RPL extension fields modified by macroinstruction

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B'1')
The partner LU accepts the already-verified indicator.

NO (B'0')
The partner LU does not accept the already-verified indicator.

CGID
Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

CONSTATE
The field in the RPL extension that indicates which state the conversation is in. It is labeled RPL6CCST in the RPL extension.
This field can have the following values for half-duplex conversations:

X'00'
RESET
X'01'
SEND
X'08'
END_CONVERSATION

This field can have the following values for full-duplex conversations:

X'00'
RESET
X'80'
FDX_RESET
X'81'
SEND/RECEIVE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

CONVSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.
NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'OB'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

SESSID
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

State changes
These changes are applicable when RCPRI indicates OK.
For half-duplex conversations, the conversation state is SEND after successful processing.
For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.
See the Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_VALID_FOR_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL_DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMH5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=CHECK

Purpose
This macroinstruction waits for completion of an asynchronous macroinstruction request and marks the RPL and RPL extension used in the request as inactive upon completion.

Usage
When asynchronous handling is specified for an RPL-based request, the application program receives control when the request has been accepted by VTAM and the requested operation has been scheduled. An APPCCMD CONTROL=CHECK macroinstruction must be issued for the RPL used for the request to determine when the macroinstruction completes and to get the return code information. APPCCMD CONTROL=CHECK cannot be issued for synchronous requests. In addition, APPCCMD CONTROL=CHECK cannot be issued for an RPL that specifies a non-APPCCMD request. This macroinstruction can be issued in cross-memory mode against an active asynchronous RPL request only when the RPL's ECB has been posted or the RPL exit has been scheduled.

When APPCCMD CONTROL=CHECK is executed for an RPL that specifies an ECB, the following actions occur:

- If the operation being checked has not been completed, VTAM suspends the execution of the application program task under which the APPCCMD CONTROL=CHECK is issued until the operation is completed.
- If the operation being checked has completed, VTAM returns control to the next sequential instruction after the APPCCMD CONTROL=CHECK macroinstruction.
- The ECB (internal or external) is cleared before VTAM returns control to the application program. (The ECB must be cleared before an RPL-based macroinstruction is issued.)

Note: The ECB specified in an asynchronous APPCCMD macroinstruction must not be cleared between the time it is issued and the corresponding APPCCMD CONTROL=CHECK is issued. If the ECB is cleared during this interval, control cannot be returned to the application program after the APPCCMD CONTROL=CHECK is issued.

- The RPL being checked is marked available for reuse by another APPCCMD macroinstruction. (APPCCMD CONTROL=CHECK is the only way this can be done for asynchronous APPCCMD requests.)

When APPCCMD CONTROL=CHECK is executed in an RPL exit routine for the associated RPL, the following actions occur:

- VTAM marks the RPL being checked as available for reuse by another APPCCMD macroinstruction.
- If the operation being checked has completed, VTAM returns control to the next sequential instruction after the APPCCMD CONTROL=CHECK.

Context
Input states are not applicable to this macroinstruction.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.
Syntax

```plaintext
APPCCMD name CONTROL = CHECK, RPL = rpl_address_field
```

Notes:

1 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

Input parameters

- **RPL=rpl_address_field**
- **RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

After the APPCCMD CONTROL=CHECK macroinstruction has completed, the completion information returned is for the macroinstruction being checked. Refer to the description of the particular APPCCMD being checked for a list of the parameters that are returned to the application program.

**APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG**

**Purpose**

This macroinstruction deallocates a conversation when an application program has detected a transaction program error.

**Usage**

QUALIFY=ABNDPROG is used when the application program detects an error in a transaction program and that error prevents further useful communication on the conversation. It corresponds to DEALLOCATE TYPE=ABEND_PROG in the LU 6.2 architecture. If the conversation is in a sending state, the SEND buffer is flushed before the conversation is deallocated.

This macroinstruction, along with the other QUALIFY=ABND* forms, can be used to cancel an outstanding APPCCMD macroinstruction, which allows the application program to recover from hung transactions. The following macroinstructions cannot be canceled:

- APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC that has not received any data from the partner LU
- APPCCMD CONTROL=RECEIVE, QUALIFY=ANY that has not been matched to a conversation
- APPCCMD CONTROL=RCVFMH5, QUALIFY=NULL|QUEUE
- APPCCMD CONTROL=RESETRCV
- APPCCMD CONTROL=OPRCNTL
- APPCCMD CONTROL=REJECT, QUALIFY=CONV
- APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL|IALL
- One of the abnormal deallocation macroinstructions
- A macroinstruction that is waiting for a response to a confirmation request
- A macroinstruction that is waiting for the arrival of an FMH-7
If any one of these macroinstructions is outstanding, the application program must wait for it to complete before issuing this macroinstruction. Alternatively, the application program can issue an APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for information on deallocating a conversation when an error is detected or for early deallocation of a pending APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5, QUALIFY=DATAQUE.

Context

On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PEND_SEND
- PEND_END_CONV_LOG
- PENDING_RECEIVE_LOG

On full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

\[
\text{APPCCMD} \quad \text{CONTROL} \quad = \quad \text{DEALLOC} \quad , \quad \text{QUALIFY} \quad = \quad \rightarrow \\

\text{ABNDPROG} \quad 1 \\

, \quad \text{RPL} \quad = \quad \text{rpl_address_field} \\
\quad (\quad \text{rpl_address_register} \quad )
\]

\[
, \quad \text{AAREA} \quad = \quad \text{rpl_extension_address_field} \\
\quad (\quad \text{rpl_extension_address_register} \quad )
\]

\[
, \quad \text{ACB} \quad = \quad \text{acb_address_field} \\
\quad (\quad \text{acb_address_register} \quad )
\]
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

**Input parameters**

The following information shows descriptions of the input parameters:
AAREA=\texttt{rpl\_extension\_address\_field}
\texttt{AAREA=(rpl\_extension\_address\_register)}
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\texttt{acb\_address\_field}
\texttt{ACB=(acb\_address\_register)}
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=\texttt{optional\_log\_data\_area\_address\_field}
\texttt{AREA=(optional\_log\_data\_area\_address\_register)}
Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the length specified in the LL field. (Refer to \textit{z/OS Communications Server: SNA Programmer’s LU 6.2 Guide} for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

\textbf{BRANCH=NO}
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

\textbf{BRANCH=YES}
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=\texttt{32\_bit\_resource\_id\_field}
\texttt{CONVID=(32\_bit\_resource\_id\_register)}
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

\textbf{ECB=INTERNAL}
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

\textbf{ECB=ecb\_address\_field}
\texttt{ECB=(ecb\_address\_register)}
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. \textit{Event\_control\_block\_address} is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=\texttt{exit\_routine\_address\_field}
\texttt{EXIT=(exit\_routine\_address\_register)}
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=optional_log_data_length
RECLEN=(optional_log_data_length_register)
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the RPL and RPL extension fields.

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'
  SEND
X'02'
  RECEIVE
X'03'
  RECEIVE_CONFIRM
X'04'
  RECEIVE_CONFIRM_SEND
X'05'
  RECEIVE_CONFIRM_DEALLOCATE
X'07'
  PENDING_END_CONVERSATION_LOG
X'08'
  END_CONVERSATION
X'09'
  PENDING_SEND
X'0A'
PENDING_RECEIVE_LOG
For full-duplex conversations, this field can have the following values:
X'80'
FDX_RESET
X'81'
SEND/RECEIVE
X'82'
SEND_ONLY
X'83'
RECEIVE_ONLY
X'84'
PENDING_SEND/RECEIVE_LOG
X'85'
PENDING_RECEIVE-ONLY_LOG
X'86'
PENDING_RESET_LOG

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LE
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**STSHBF**
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**State changes**
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful completion of the macroinstruction. For full-duplex conversations, the conversation state is FDX_RESET after successful completion of the macroinstruction.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV

Purpose
This macroinstruction is used when the application program detects an error in its implementation of LU 6.2 services.

Usage
QUALIFY=ABNDSERV is used when the application program encounters errors related to LU 6.2 services. For example, the application program might detect an error in its support of mapped conversations or in conversation-level security that would require it to deallocate the conversation. QUALIFY=ABNDSERV corresponds to the DEALLOCATE TYPE=ABEND_SVC verb in the LU 6.2 architecture.

If the conversation is in a state that allows sending, the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction is executed prior to deallocating the conversation.

APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV can be issued against a conversation for which there is already an APPCCMD outstanding. These commands cancel the previous macroinstruction, allowing the application program to recover from a "hung" transaction. However, there are cases where it is not allowed when a prior macroinstruction is outstanding. See “Usage” on page 60 for a list of macroinstructions that cannot be canceled.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for information on abnormally deallocating a conversation.

Context
On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
On full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD — — CONTROL — — DEALLOC — — QUALIFY — —

ABNDsrv 1

RPL  — — rpl_address_field
     ( rpl_address_register )

AAREA  — — rpl_extension_address_field
        ( rpl_extension_address_register )

ACB  — — acb_address_field
      ( acb_address_register )

AREA  — — optional_log_data_area_address_field
       ( optional_log_data_area_address_register )

BRANCH  = NO 3
         YES

CONVID  = 32-bit_resource_id_field
         ( 32-bit_resource_id_register )
```
Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**  
**AAREA=(rpl_extension_address_register)**  
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**  
**ACB=(acb_address_register)**  
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=optional_log_data_area_address_field

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the length specified in the LL field. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field

CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=optional_log_data_length
RECLEN=(optional_log_data_length_register)
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the PRL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'
SEND

X'02'
RECEIVE

X'03'
RECEIVE_CONFIRM

X'04'
RECEIVE_CONFIRM_SEND

X'05'
RECEIVE_CONFIRM_DEALLOCATE

X'07'
PENDING_END_CONVERSATION_LOG

X'08'
END_CONVERSATION

X'09'
PENDING_SEND

X'0A'
PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

X'80'
FDX_RESET

X'81'
SEND/RECEIVE
**EXPDLLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B’1’)**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFHM5 in order to receive an FMH-5.

**NO (B’0’)**
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**STSHBF**
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while
data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STD in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**State changes**
The changes are applicable when RCPRI indicates OK.
For half-duplex conversations, the conversation state is END_CONV after successful processing.
For full-duplex conversations, the conversation state is FDX_RESET after successful processing.
See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>PARAMETER_ERROR—ABNORMAL_DEALLOCATE_REJECTED_RETRY</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME**

**Purpose**

This macroinstruction is used to deallocate a conversation that has had no activity for a specified amount of time.

**Usage**

QUALIFY=ABNDTIME is used when the LU detects that it has not received information from one of its transaction programs within a specific amount of time. For example, an application program would use this macroinstruction if one of the conversations is in a state that allows receiving and has not received any data in an excessive amount of time. The application program must determine how long to wait before issuing this macroinstruction.

If the conversation is in a state that allows sending, the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction is executed prior to abnormally deallocating the conversation.

APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME can be issued against a conversation for which there is already an APPCCMD outstanding. These commands cancel the previous macroinstruction, allowing the application program to recover from a hung transaction. However, there are cases where it is not allowed when a prior macroinstruction is outstanding. See “Usage” on page 60 for a list of macroinstructions that cannot be canceled.

QUALIFY=ABNDTIME corresponds to the DEALLOCATE TYPE=ABEND_TIMER verb in the LU 6.2 architecture.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of abnormally terminating a conversation.

**Context**

On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PEND_SEND
- PEND_END_CONV_LOG
- PENDING_RECEIVE_LOG
On full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```plaintext
name APPCCMD  --- CONTROL --- DEALLOC --- QUALIFY ---
   ABNDTIME 1

, --- RPL = rpl_address_field
   (— rpl_address_register —)

, --- AAREA = rpl_extension_address_field
   (— rpl_extension_address_register —)

, --- ACB = acb_address_field
   (— acb_address_register —)

, --- AREA = optional_log_data_area_address_field
   (— optional_log_data_area_address_register —)

, --- BRANCH = NO 3
   YES

, --- CONVID = 32-bit_resource_id_field
   (— 32-bit_resource_id_register —)
```
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=**\texttt{rpl\_extension\_address\_field}

AAREA=\texttt{(rpl\_extension\_address\_register)}

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**\texttt{acb\_address\_field}

ACB=\texttt{(acb\_address\_register)}

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=optional_log_data_area_address_field
AREA=(optional_log_data_area_address_register)

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the
partner application program. The application program is responsible for placing the error log data into
the local system log. If the application program chooses to supply an error log GDS variable, it has to
supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-
record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the
length specified in the LL field. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2
Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in
supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use
BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field
of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state
(non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather
than under a TCB, the macroinstruction is processed in this manner automatically, regardless of
the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the
completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single
APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD
completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be
any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD
completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPLEXTDS field of the RPL.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the
function of the APPCCMD has completed. The indicator resides within the RPL.

OPTCD=ASY

Specifies that control is to be returned to the application program immediately and that the
application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPL.

LU 6.2 macroinstruction syntax and operands 77
OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=optional_log_data_length
RECLEN=(optional_log_data_length_register)
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6AVFA in the RPL extension.
For half-duplex conversations, this field can have the following values:
X'01'
SEND
X'02'
RECEIVE
X'03'
RECEIVE_CONFIRM
X'04'
RECEIVE_CONFIRM_SEND
X'05'
RECEIVE_CONFIRM_DEALLOCATE
X'07'
PENDING_END_CONVERSATION_LOG
X'08'
END_CONVERSATION
X'09'
PENDING_SEND
X'0A'
PENDING_RECEIVE_LOG
For full-duplex conversations, this field can have the following values:
X'80'
FDX_RESET
X'81'
SEND/RECEIVE
**EXPDLLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B’1’)**
  
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B’0’)**
  
  No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**STSHBF**

The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while
data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**State changes**
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful processing.

For full-duplex conversations, the conversation state is FDX_RESET after successful processing.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>PARAMETER_ERROR—ABNORMAL_DEALLOCATE_REJECTED_RETRY</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER**

**Purpose**

This macroinstruction deallocates a conversation when the application program detects an error.

**Usage**

This macroinstruction is used by an application program to deallocate a conversation and to inform the partner LU of the reason for the deallocation. To indicate the reason for the deallocation, the application program specifies a sense code on the macroinstruction. This sense code is sent to the partner LU in an FMH-7 and must be appropriate to the error. Otherwise improper processing of the macroinstruction might occur. For a list of valid sense codes, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

This macroinstruction does not correspond to any of the verbs in the LU 6.2 architecture.

An example of the use of this macroinstruction would be to report errors that the application program detects on a received FMH-5. Although VTAM performs preliminary format checks on the FMH-5 before passing it to the application program, the application program validates the FMH-5. If the application program detects an error in the FMH-5, it issues APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER and specifies the appropriate sense code. VTAM sends the conversation deallocation notification and the FMH-7 to the partner LU.

If the conversation is in a state that allows sending, the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction is executed prior to abnormally deallocating the conversation.

APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER can be issued against a conversation for which there is already an APPCCMD outstanding. It cancels the previous macroinstruction, allowing the application program to recover from a "hung" transaction. However, in some cases, it is not allowed when a prior macroinstruction is outstanding. See “Usage” on page 60 for a list of macroinstructions that cannot be canceled.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information on abnormally deallocating a conversation.

**Context**

On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
• RECEIVE_CONFIRM_SEND
• RECEIVE_CONFIRM_DEALLOCATE
• PEND_SEND
• PEND_END_CONV_LOG
• PENDING_RECEIVE_LOG

On full-duplex conversations, this macroinstruction can be issued from the following conversation states:
• PENDING_ALLOCATE
• SEND/RECEIVE
• SEND_ONLY
• RECEIVE_ONLY
• PENDING_SEND/RECEIVE_LOG
• PENDING_RECEIVE-ONLY_LOG
• PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

\[
\text{APPCCMD} \quad \text{CONTROL} \quad = \quad \text{DEALLOC} \quad , \quad \text{QUALIFY} \quad = \quad \text{ABNDUSER}^1, \\
\quad \quad \text{RPL} \quad = \quad rpl\_address\_field \quad \quad \text{(} \quad rpl\_address\_register \quad ) \quad \quad \text{3} \quad \\
\quad \quad \text{AAREA} \quad = \quad rpl\_extension\_address\_field \quad \quad \text{(} \quad rpl\_extension\_address\_register \quad ) \quad \quad \text{3} \quad \\
\quad \quad \text{ACB} \quad = \quad acb\_address\_field \quad \quad \text{(} \quad acb\_address\_register \quad ) \quad \quad \text{3} \quad \\
\quad \quad \text{AREA} \quad = \quad \text{optional}_\text{log}\_\text{data}\_area\_address\_field \quad \quad \text{(} \quad \text{optional}_\text{log}\_\text{data}\_area\_address\_register \quad ) \quad \quad \text{3} \quad \\
\quad \quad \text{BRANCH} \quad = \quad \text{NO} \quad \quad \text{YES} \quad \quad \text{3} \quad \\
\quad \quad \text{CONVID} \quad = \quad 32\text{-bit}\_\text{resource}\_\text{id}\_\text{field} \quad \quad \text{(} \quad 32\text{-bit}\_\text{resource}\_\text{id}\_\text{register} \quad ) \quad \quad \text{1} \quad 
\]
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
   Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
   Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with
transaction programs. Application programs cannot issue APPCCMD macroinstructions in address
spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=optional_log_data_area_address_field
AREA=(optional_log_data_area_address_register)
Specifies the address of a data area containing a formatted error log GDS variable to be sent to the
partner application program. The application program is responsible for placing the error log data into
the local system log. If the application program chooses to supply an error log GDS variable, it has to
supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-
record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the
length specified in the LL field. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2
Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in
supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use
BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of
the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state
(non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather
than under a TCB, the macroinstruction is processed in this manner automatically, regardless of
the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the
completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single
APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD
completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be
any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD
completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the
function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the
RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the
application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of
the RPL.
**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=optional_log_data_length**

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

**RPL=rpl_address_field**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**SENSE=user-supplied_32-bit_fmh-7_sense_code**

Specifies the user-specified sense code that the application program requests to be placed in the FMH-7 that VTAM creates as a result of this APPCCMD macroinstruction. This sense code must be appropriate to the error. Otherwise, improper processing of the macroinstruction might result. This is the only one of the abnormal DEALLOC macroinstructions for which this field is applicable. This field is labeled RPL6SNSO in the RPL extension. For a list of valid sense codes, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- **'X'01'**
  - SEND
- **'X'02'**
  - RECEIVE
- **'X'03'**
  - RECEIVE_CONFIRM
- **'X'04'**
  - RECEIVE_CONFIRM_SEND
- **'X'05'**
  - RECEIVE_CONFIRM_DEALLOCATE
- **'X'07'**
  - PENDING_END_CONVERSATION_LOG
- **'X'08'**
  - END_CONVERSATION
- **'X'09'**
  - PENDING_SEND
For full-duplex conversations, this field can have the following values:

- **X'80'**
  - FDX_RESET
- **X'81'**
  - SEND/RECEIVE
- **X'82'**
  - SEND_ONLY
- **X'83'**
  - RECEIVE_ONLY
- **X'84'**
  - PENDING_SEND/RECEIVE_LOG
- **X'85'**
  - PENDING_RECEIVE-ONLY_LOG
- **X'86'**
  - PENDING_RESET_LOG

**EXPDLLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RLP in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  - One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**
  - No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

State changes
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful processing.
For full-duplex conversations, the conversation state is FDX_RESET after successful processing.
See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=DEALLOC, QUALIFY=CONFIRM

Purpose
This macroinstruction sends a confirmation request to a partner application program and, if the partner sends a positive confirmation response, VTAM deallocates the conversation.

Usage
QUALIFY=CONFIRM is used to ensure that the partner receives all data on a conversation before that conversation is deallocated.

VTAM sends the partner LU any remaining data in the SEND buffer, which is followed by a confirmation request. If the partner LU sends a positive response to the confirmation request, VTAM deallocates the conversation. If the partner LU sends a negative response to the confirmation request, VTAM does not deallocate the conversation. This macroinstruction completes only after a response is received from the partner LU. It corresponds to the DEALLOCATE (TYPE=CONFIRM) verb in the LU 6.2 architecture.

When this macroinstruction completes, the current conversation state is in the CONSTATE field.

Because this macroinstruction requests deallocation of the conversation, the data in the SEND buffer must complete a logical record.

For more information on sending and responding to confirmation requests, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

Context
This macroinstruction can only be used on half-duplex conversations from the SEND conversation state.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>PARAMETER_ERROR—ABNORMAL_DEALLOCATE_REJECTED_RETRY</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002D'</td>
<td>PARAMETER_ERROR—INVALIDSENSE_CODE_VALUE_SPECIFIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_Resource_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field

 Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.
**CONMODE=BUFFCA**
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**
**CONVID=(32-bit_resource_id_register)**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**
**ECB=(ecb_address_register)**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.
EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. It can have the following values:

X'01'
    SEND
X'02'
    RECEIVE
X'03'
    RECEIVE_CONFIRM
X'04'
    RECEIVE_CONFIRM_SEND
X'05'
    RECEIVE_CONFIRM_DEALLOCATE
X'07'
    PENDING_END_CONVERSATION_LOG
X'08'
    END_CONVERSATION
X'09'
    PENDING_SEND
X'0A'
  PENDING_RECEIVE_LOG

EXPDLEN
  The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
  The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
  The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
  The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
  The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
  No FMH-5s are waiting to be received by the application program.

LOGRCV
  The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
  Indicates that an FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
  ALLOCATION_ERROR

X'0014'
  DEALLOCATE_ABEND_PROGRAM

X'0018'
  DEALLOCATE_ABEND_SERVICE

X'001C'
  DEALLOCATE_ABEND_TIMER

X'0030'
  PROGRAM_ERROR_NO_TRUNC

X'0034'
  PROGRAM_ERROR_PURGING
NO (B’0’)
Indicates either that no error indicator was received or that an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that VTAM did not recognize. This field is labeled RPL6SNSI in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
These changes are applicable when RCPRI indicates OK.
The conversation state is END_CONV after successful processing.
See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.
<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=DEALLOC, QUALIFY=DATACON

#### Purpose
This macroinstruction sends data, which is supplied by the application program, and any information in the SEND buffer to a partner application program, followed by a confirmation request. If the partner LU sends a positive response to the confirmation request, VTAM deallocates the conversation normally.

#### Usage
This macroinstruction is used to send data to the partner LU and to ensure that the partner receives all the data before the conversation is deallocated.

VTAM sends any data remaining in the buffer followed by the data specified on the macroinstruction to the partner LU. This data is followed by a confirmation request. The macroinstruction completes only after the partner LU responds to the confirmation request. If the partner sends a positive confirmation response, the conversation is deallocated. If the partner LU sends a negative confirmation response, the conversation is not deallocated. This macroinstruction corresponds to the SEND_DATA and DEALLOCATE (TYPE=CONFIRM) verbs in the LU 6.2 architecture.

When this macroinstruction completes, the current conversation state is found in the CONSTATE field.

Because this macroinstruction requests deallocation of the conversation, the data sent must complete a logical record.

For more information on sending and responding to confirmation requests, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

#### Context
This macroinstruction can be used only on half-duplex conversations from the SEND conversation state. This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

#### Syntax
```
APPCCMD  — — CONTROL — = — DEALLOC — , — QUALIFY — = —
       — — DATACON 1
           — — RPL — = — rpl_address_field
               ( — — rpl_address_register — — )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AREA=(data_area_or_buffer_list_address_register)**

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this
conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=XBULST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.
CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X’002C’, X’0010’ (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

RECLLEN=\text{data\_length} RECLLEN=(\text{data\_length\_register})
Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLLEN in the RPL.

- If OPTCD=NBUFFLST, RECLLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

LU 6.2 macroinstruction syntax and operands 101
RPL=rpl_address_field
RPL=(rpl_address_register)
   Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
   The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.
   This field can have the following values:
      X'01'
       SEND
      X'02'
       RECEIVE
      X'03'
       RECEIVE_CONFIRM
      X'04'
       RECEIVE_CONFIRM_SEND
      X'05'
       RECEIVE_CONFIRM_DEALLOCATE
      X'07'
       PENDING_END_CONVERSATION_LOG
      X'08'
       END_CONVERSATION
      X'09'
       PENDING_SEND
      X'0A'
       PENDING_RECEIVE_LOG

EXPDLLEN
   The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
   The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
   The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
   The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
   The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B’1’)
   One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application
The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**LOGRCV**
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**YES (B'1')**
Indicates that an FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004'  ALLOCATION_ERROR
- X'0014'  DEALLOCATE_ABEND_PROGRAM
- X'0018'  DEALLOCATE_ABEND_SERVICE
- X'001C'  DEALLOCATE_ABEND_TIMER
- X'0030'  PROGRAM_ERROR_NO_TRUNC
- X'0034'  PROGRAM_ERROR_PURGING
- X'0038'  PROGRAM_ERROR_TRUNC
- X'003C'  SERVICE_ERROR_NO_TRUNC
- X'0040'  SERVICE_ERROR_PURGING
- X'0044'  SERVICE_ERROR_TRUNC
- X'005C'  USER_ERROR_CODE_RECEIVED

**NO (B'0')**
Indicates either that no error indicator was received or that an error indicator was received but indicated that no log data follows.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM. This field is labeled RPL6SNSI in the RPL extension.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. It is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether the application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). This field is labeled RPL6RSIG in the RPL extension.

YES (B'1')
Indicates that a SIGNAL RU has been received from the partner application program. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
Indicates that no SIGNAL RU has been received from the partner application program. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.
STSHDS

The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

State changes

These changes are applicable when RCPRI indicates OK.

The conversation state is END_CONV after successful processing.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED_FOLLWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED_CONTROLQUALIFY_VALUE_NOT_VALID_FORFULL-DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR_SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR_RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR_NOT_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0002'</td>
<td>CSM_DETECTED_ERROR_INVALID_BUFFER_TOKEN_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0003'</td>
<td>CSM_DETECTED_ERROR_INVALID_INSTANCE_ID_SPECIFIED</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=DEALLOC, QUALIFY=DATAFLU

Purpose
This macroinstruction unconditionally deallocates a conversation after sending data to a partner application program. The send function of the macroinstruction includes flushing the SEND buffer.

Usage
This macroinstruction combines the functions of two macroinstructions, APPCCMD CONTROL=SEND, QUALIFY=DATA followed by APPCCMD CONTROL=DEALLOC, QUALIFY=FLUSH. As with all macroinstructions that both send data and deallocate a conversation, the data sent by the application program must complete a logical record.

The deallocation request on this macroinstruction is unconditional. After VTAM successfully sends the data, it deallocates the conversation. Any incoming error information received for the application program is discarded.

This macroinstruction corresponds to the SEND_DATA verb followed by the DEALLOCATE (TYPE=FLUSH) verb described in the LU 6.2 architecture.

Context
For half-duplex conversations, this macroinstruction can be issued from the SEND or PENDING_SEND conversation states.

For full-duplex conversations, this macroinstruction can be issued from the following states:
• SEND/RECEIVE
• SEND_ONLY
• PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
name

APPCCMD  CONTROL  =  DEALLOC  ,  QUALIFY  =  

DATAFLU  1

RPL  =  rpl_address_field

(  rpl_address_register  )

AAREA  =  rpl_extension_address_field

(  rpl_extension_address_register  )

ACB  =  acb_address_field

(  acb_address_register  )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=data_area_or_buffer_list_address_field**

AREA=(data_area_or_buffer_list_address_register)

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains
additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLEXTDS field of the RPL.
**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=BUFFLST**
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

**OPTCD=NBUFFLST**
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

**OPTCD=XBUFLST**
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X’002C’, X’0010’ (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

**RECLEN=**

**RECLEN=(data_length_register)**
Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

**RPL=**

**RPL=(rpl_address_register)**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension. For half-duplex conversations, this field can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80' FDX_RESET
- X'81' SEND/RECEIVE
- X'82' SEND_ONLY
- X'83' RECEIVE_ONLY
- X'84' PENDING_SEND/RECEIVE_LOG
- X'85' PENDING_RECEIVE-ONLY_LOG
- X'86' PENDING_RESET_LOG

**EXPDLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.
FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMHS set on or off). This field is labeled RPL6RMHS in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application’s responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
Returns the sense code carried in the FMH-7 used in deallocating the conversation. This field is labeled RPL6SNSI in the RPL extension.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. It is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.
SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner
has requested permission to send. This field and the SIGDATA field correspond to the
REQUEST_TO_SEND RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation
(APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified
RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). This field is labeled RPL6RSIG in the
RPL extension.

**YES (B'1')**
Indicates that a SIGNAL RU has been received from the partner application program. The values
carried in the signal code and signal extension fields of the SIGNAL RU are returned to the
application program in the SIGDATA field.

**NO (B'0')**
Indicates that no SIGNAL RU has been received from the partner application program. When
SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to
give the current position (address and displacement) in the application-supplied data buffer or buffer
list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while
data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL
extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with
STSHBF to give the current position (address and displacement) in the application-supplied data
buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage
shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is
labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a
conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL
extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the
conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5
macroinstruction (if the conversation was initiated by a remote application program). This field is
labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

State changes
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation enters END_CONV after successful completion of the
macroinstruction.

For full-duplex conversations, the conversation enters one of the following states after successful
completion of the macroinstruction.

- RECEIVE ONLY
- PENDING RECEIVE ONLY LOG
- FDX RESET

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.
### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>SYNC_LEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>TRANSACTION_PROGRAM_NOT_AVAILABLE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>TRANSACTION_PROGRAM_NOT_AVAILABLE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>CANNOT_RECONNECT_TRANSACTION_PROGRAM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>CANNOT_RECONNECT_TRANSACTION_PROGRAM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>RECONNECT_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNCATION</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNCATING</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNCATION</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>INVALID_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOTAUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00AB'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>ERROR_INDICATION_RECEIVED_DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>ERROR_INDICATION_RECEIVED_DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>ERROR_INDICATION_RECEIVED_DEALLOCATE_ABEND_TIME</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>ERROR_INDICATION_RECEIVED_ALLOCATION_ERROR</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>ERROR_INDICATION_RECEIVED_UNKNOWN_ERROR_CODE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>ERROR_INDICATION_RECEIVED_Resource_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>ERROR_INDICATION_RECEIVED_Resource_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0002'</td>
<td>CSM_DETECTED_ERROR—INVALID_BUFFER_TOKEN_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0003'</td>
<td>CSM_DETECTED_ERROR—INVALID_INSTANCE_ID_SPECIFIED</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=DEALLOC, QUALIFY=FLUSH**

**Purpose**  
This macroinstruction flushes the SEND buffer and unconditionally deallocates a conversation.

**Usage**  
For half-duplex conversations, this macroinstruction executes the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction prior to the deallocation. Any error information coming from the partner application program that is received by VTAM after the macroinstruction is issued is not reported to the application program.
This macroinstruction, when issued on a full-duplex conversation, either initiates the conversation deallocation or completes the conversation deallocation if a deallocation request has been received from the conversation partner.

This macroinstruction corresponds to the DEALLOCATE (TYPE=FLUSH) verb described in the LU 6.2 architecture.

**Context**

For half-duplex conversations, this macroinstruction can be issued from a SEND or PENDING_SEND conversation state.

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```plaintext
name APPCCMD CONTROL DEALLOC, QUALIFY = FLUSH

, RPL = rpl_address_field
  ( rpl_address_register )

, AAREA = rpl_extension_address_field
  ( rpl_extension_address_register )

, ACB = acb_address_field
  ( acb_address_register )

, BRANCH = NO
  YES

, CONMODE = BUFFCA
  CS
  LLCA
  SAME

, CONVID = 32-bit_resource_id_field
  ( 32-bit_resource_id_register )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** `rpl_extension_address_field`

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** `acb_address_field`

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.
ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPC/CM macroinstruction. You cannot specify both ECB and EXIT on a single APPC/CM macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPC/CM macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPC/CM completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPC/CM completes. You cannot specify both ECB and EXIT on a single APPC/CM macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPC/CM has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPC/CM macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

`X'01'`
- SEND

`X'02'`
- RECEIVE
For full-duplex conversations, this field can have the following values:

**EXPDLN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMHSLN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program.
program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCMC completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

State changes
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation enters END_CONV after successful completion of the macroinstruction.

For full-duplex conversations, the conversation can enter the following conversation states after successful processing:

• RECEIVE_ONLY
• PENDING_RECEIVE-ONLY_LOG
• FDX_RESET

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=DEALLOCQ

**Purpose**
This macroinstruction deallocates a conversation when an application program has detected an error. This macroinstruction is queued if the conversation is in the RECEIVE state and has not yet received data. When data is received, VTAM continues deallocation of the conversation.

**Usage**
QUALIFY=ABNDPROG is used to abnormally terminate a conversation when the application program detects an error that will prevent further useful conversation.

QUALIFY=ABNDSERV is used to abnormally terminate a conversation and alert VTAM that an LU service component has encountered an error.

QUALIFY=ABNDTIME is used to abnormally terminate a conversation when the application program detects that it has not received information from its partner for a specified amount of time.
QUALIFY=ABNDUSER is used to abnormally terminate a conversation. The command also alerts VTAM that the application program will provide a user-specified sense code to place in the FMH-7 that VTAM creates as a result of this command. The application program is responsible for the validity of the sense code.

This macroinstruction abnormally deallocates a conversation. If the conversation is in a sending state, the function is identical to the abnormal termination APPCCMD CONTROL=DEALLOC. The SEND buffer is flushed before the conversation is deallocated.

If the conversation is in a receiving state and is waiting for a first, or only element in the chain, this macroinstruction is queued until data is received from the partner LU.

To contrast this macroinstruction with DEALLOC, the DEALLOCQ macroinstruction will never receive an RCPRI, RCSEC of X'002C', X'0021'.

The following macroinstructions cannot be canceled by APPCCMD CONTROL=DEALLOCQ:

- APPCCMD CONTROL=RECEIVE, QUALIFY=ANY that has not been matched to a conversation
- APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY that has not been matched to a conversation
- APPCCMD CONTROL=RCVFMH5, QUALIFY=NULL|QUEUE
- APPCCMD CONTROL=RESETRCV
- APPCCMD CONTROL=OPRCNTL
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG|ABNDSERV|ABNDTIME|ABNDUSER
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDPROG|ABNDSERV|ABNDTIME|ABNDUSER
- APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL|IALL
- A macroinstruction that is waiting for a response to a confirmation request
- A macroinstruction that is waiting for the arrival of an FMH-7

If any one of these macroinstructions is outstanding, the application program can either wait for the outstanding APPCCMD to complete and then issue APPCCMD CONTROL=DEALLOCQ or issue APPCCMD CONTROL=REJECT.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for information on deallocating a conversation when an error is detected.

For early deallocation of a pending APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5, QUALIFY=DATAQUE, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PENDING_SEND
- PENDING_END_CONV_LOG
- PENDING_RECEIVE_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
APPCCMD — — CONTROL — — DEALLOCQ — — QUALIFY — —

ABNDPROG
ABNDSERV
ABNDTIME
ABNDUSER

RPL — — rpl_address_field — — ( rpl_address_register — )

AAREA — — rpl_extension_address_field — — ( rpl_extension_address_register — )

ACB — — acb_address_field — — ( acb_address_register — )

AREA — — data_area_or_buffer_list_address_field — — ( data_area_or_buffer_list_address_register — )

BRANCH — — NO — — YES

CONVID — — 32-bit_resource_id_field — — ( 32-bit_resource_id_register — )

ECB — — INTERNAL — —

ecb_address_field — — ( ecb_address_register — )

EXIT — — exit_routine_address_field — — ( exit_routine_address_register — )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\textbf{AAREA=\textit{rpl\_extension\_address\_field}}
\textbf{AAREA=(\textit{rpl\_extension\_address\_register})}

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\textbf{ACB=\textit{acb\_address\_field}}
\textbf{ACB=(\textit{acb\_address\_register})}

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

\textbf{AREA=\textit{optional\_log\_data\_area\_address\_field}}
\textbf{AREA=(\textit{optional\_log\_data\_area\_address\_register})}

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the
LENGTH specified in the LL field. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
RECLEN=optional_log_data_length
RECLEN=(optional_log_data_length_register)
   Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

RPL=rpl_address_field
RPL=(rpl_address_register)
   Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SENSE
   Specifies the user-specified sense code that the application program requests to be placed in the FMH-7 that VTAM creates as a result of this APPCCMD macroinstruction. This sense code must be appropriate to the error. Otherwise, improper processing of the macroinstruction might result. This field is examined only if QUALIFY=ABNDUSER is issued. This field is labeled RPL6SNSI in the RPL extension. For a list of valid sense codes,

RPL and RPL extension fields modified by macroinstruction
   The following information shows descriptions of the RPL and RPL extension fields:

CONSTATE
   The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.
   For half-duplex conversations, this field can have the following values:
   X'01'
      SEND
   X'02'
      RECEIVE
   X'03'
      RECEIVE_CONFIRM
   X'04'
      RECEIVE_CONFIRM_SEND
   X'05'
      RECEIVE_CONFIRM_DEALLOCATE
   X'07'
      PENDING_END_CONVERSATION_LOG
   X'08'
      END_CONVERSATION
   X'09'
      PENDING_SEND
   X'0A'
      PENDING_RECEIVE_LOG
   For full-duplex conversations, this field can have the following values:
   X'80'
      FDX_RESET
   X'81'
      SEND/RECEIVE
   X'82'
      SEND_ONLY
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLENS specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

No FMH-5s are waiting to be received by the application program.

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RPCR in the RPL extension.

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.
**STSHDS**

The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

**State changes**

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, END_CONV state is entered.

For full-duplex conversations, FDX_RESET state is entered.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002D'</td>
<td>PARAMETER_ERROR—INVALIDSENSE_CODE_VALUE_SPECIFIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTEDVECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS

Purpose

This macroinstruction responds positively to a session establishment request.

Usage

This macroinstruction is issued after the application program is notified through its LOGON or SCIP exit routine that a CINIT or BIND request has been received. (For a description of when the LOGON and SCIP exits are scheduled and for the information provided in each exit, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.) The function of this command is similar to the VTAM API commands OPNDST OPTCD=ACCEPT and OPNSEC for non-LU 6.2 sessions.

When this macroinstruction is used in a LOGON exit, the RPLAREA field of the read-only RPL passed to the exit routine contains the address of a read-only copy of the CINIT. The application program can examine the parameters of the BIND in the CINIT. If the application program needs to override any of the BIND parameters, it can specify session parameters for a BIND on this macroinstruction (mapped by ISTDBIND).

Attention: If both the local and the partner LU are the same LU, then this macroinstruction must not be issued from the LOGON exit routine. Otherwise, the session will hang.

The partner LU can negotiate the BIND. If this occurs, VTAM verifies and accepts the negotiated BIND parameters. (For information on BIND fields and their settings, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.) However, VTAM does not return the negotiated BIND response to the application program when this macroinstruction completes.

The application program can use this macroinstruction in a SCIP exit to override some of the values received in the BIND by providing a BIND image (in ISTDBIND format) to be used in building a response. When this macroinstruction is used in a SCIP exit, word 4 of the parameter list points to session parameters mapped by ISTDBIND. If the application program needs to override any of the BIND parameters, it can specify session parameters for a BIND response on this macroinstruction (mapped by ISTDBIND). Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for details on the values that can be overridden.

Note: APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS does not correspond to the ACTIVATE_SESSION verb described in the LU 6.2 architecture.

Context

Input states are not applicable to this macroinstruction.

Syntax

```
APPCCMD — CONTROL — OPRCNTL — QUALIFY — 

ACTSESS

RPL — rpl_address_field

AAREA — rpl_extension_address_field

ACB — acb_address_field

AREA — session_parameter_address_field

ARG — 4-byte_session_identifier_(cid)_field

BRANCH — NO

CONFTXT — NO

ECB — INTERNAL

EXIT — exit_routine_address_field

LU 6.2 macroinstruction syntax and operands 133
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=rpl_extension_address_field

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=acb_address_field

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

**AREA**=session_parameter_address_field

Specifies the address of an area that contains a set of session parameters that VTAM uses when constructing the BIND or BIND response, which is sent to establish a session. If an address is indicated, the set of parameters specified by the application program will override the session parameters given in the CINIT or BIND (refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for information on building the session parameters). This field is labeled RPLAREA in the RPL.
RPL. If you specify AREA=0, VTAM uses the set of session parameters contained in the CINIT or BIND to construct the BIND or BIND response.

**Note:** You should use the ISTDBIND DSECT if you include user data fields on the BIND.

**ARG=4-byte_session_identifier_(cid)_field**

Specifies the CID of the session that was returned to the application program in the parameter list of the LOGON or SCIP exit routine. The specified CID must identify a CINIT or BIND that is queued for this application program.

**ARG=(4-byte_session_identifier_(cid)_register)**

Specifies the CID of the session that was returned to the application program in the parameter list of the LOGON or SCIP exit routine. The specified CID must identify a CINIT or BIND that is queued for this application program.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONFTXT**

Indicates whether or not data sent or received on this session is to be considered "confidential" within this host. This field is labeled RPL6CFTX in the RPL extension.

**CONFTXT=YES**

The VTAM buffers used to hold the data are cleared before they are returned to their buffer pools.

**CONFTXT=NO**

No clearing is performed.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=session_parameter_information_length**

**RECLEN=(session_parameter_information_length_register)**

Specifies the length of the session parameter information. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**USERFLD=4_bytes_of_user_data**

**USERFLD=(user_data_register)**

Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information descriptions of RPL and RPL extension fields:

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

Contains the sense code if any is returned from session initiation macroinstructions. This field is labeled RPL6SNSI in the RPL extension.

**USERFLD**

Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.
### APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS

**Purpose**

This macroinstruction negotiates the session limits on a mode group between the application program and a partner application.

**Usage**

VTAM determines the new session limits by using the session limits specified on the macroinstruction and the defined session limits of the partner LU. The overall session limits, the contention-winner session limits, and the contention-loser session limits are negotiated. Other parameters, such as draining of a conversation request and responsibility for deactivation, are also negotiated by this macroinstruction.

When this macroinstruction completes, VTAM can activate or deactivate sessions to make them conform to the new session limits. However, sessions already assigned to a conversation are not deactivated.

This macroinstruction corresponds to the INITIALIZE_SESSION_LIMIT, CHANGE_SESSION_LIMIT, and RESET_SESSION_LIMIT verbs in the LU 6.2 architecture.

The APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE macroinstruction can be used by a partner LU that is capable of parallel sessions to define the session limits that can be used in the negotiation when it receives the CNOS request.
For a full discussion of this macroinstruction, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**Context**

Input states are not applicable to this macroinstruction.

When a mode is retained for persistent LU-LU sessions, the QUALIFY=CNOS macroinstruction is not allowed.

**Syntax**

```
name APPCCMD CONTROL OPRCNTL QUALIFY CNOS
RPL rpl_address_field (rpl_address_register)
AAREA rpl_extension_address_field (rpl_extension_address_register)
ACB acb_address_field (acb_address_register)
AREA cnos_session_limits_structure_address_field (cnos_session_limits_structure_address_register)
BRANCH NO YES
ECB INTERNAL ecb_address_field (ecb_address_register)
EXIT exit_routine_address_field (exit_routine_address_register)
LOGMODE 8-byte_logon_mode_name
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=

```
AAREA=(rpl_extension_address_register)
```

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=

```
ACB=(acb_address_register)
```

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**=

```
AREA=(cnos_session_limits_structure_address_register)
```

Specifies the address of a data area containing a CNOS session limits data structure. (See “CNOS session limits data structure (ISTSLCNS)” on page 586 for the VTAM-supplied DSECT that can be used to fill in and test values.) The specification of a session limits structure is optional (the AREA field in the RPL extension would be 0 in this case). The defaults that are used when a session limits structure is omitted are given in the description of each parameter. The fields in the data structure that apply to this macroinstruction are described in the z/OS Communications Server: SNA Programmer's LU 6.2 Guide. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

```
ECB=(ecb_address_register)
```

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

```
EXIT=(exit_routine_address_register)
```

This field is labeled RPLEXTDS in the RPL.

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.
LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name that requires the session limit and contention-winner polarity values to be changed. The mode name specified can be any mode name that is valid as the LOGMODE value on the APPCCMD CONTROL=ALLOC macroinstruction including the SNASVCMG mode name, which is used for exchanging the CNOS request and reply when the application program and partner application are connected by parallel sessions. However, no CNOS flow occurs to the partner application program as a result of issuing this macroinstruction for the SNASVCMG mode name.

If the session limits control block specifies that SESSLIM=0 and NBRMODE=ALL, the session limit negotiation applies to all noncontrol modes between the two LUs, and this parameter is ignored.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is less than 8 characters, VTAM pads it on the right with blanks. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information). This field is labeled RPL6MODE in the RPL extension.

LUAFFIN
Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

LUAFFIN=APPL
The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL
VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

LUNAME=8-byte_lu_name
Specifies the name of the partner application program to which the change in the session limit and contention-winner polarity values applies. The LU name is a name that is valid as the LU name value on the APPCCMD CONTROL=ALLOC macroinstruction.

The LU name can be up to 8 characters in length. If it is less than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NAMEUSE
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

NAMEUSE=APNAME
The application identifies itself to the partner LU by its application network name.

NAMEUSE=GNAME
The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program to which the change in the session limit and contention-winner polarity value applies.
If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.) The network identifier is also used to verify and update the logon mode table. It is the same as the NETID value on the APPCCMD CONTROL=ALLOC macroinstruction.

The network identifier can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=cnos_session_limits_structure_length**

Specifies the length of the CNOS session limits data structure supplied by the AREA field. The application program must supply the entire session limits data structure; it cannot supply a partial structure. This field is applicable only if a CNOS session limits structure is specified by the AREA field. Otherwise, it is ignored by VTAM. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**USERFLD=4_bytes_of_user_data**

Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

**VTRINA=vector_address_field**

Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.
This field is labeled RPL6VAIA in the RPL extension.

**VTRINL=vector_length_field**

**VTRINL=(vector_length_register)**

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**AVFA**

The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

**YES (B'1')**

The partner LU accepts the already-verified indicator.

**NO (B'0')**

The partner LU does not accept the already-verified indicator.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**LUAFFIN**

The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

**NONE (B'00')**

GR affinity is not applicable or is unknown.

**NOTAPPL (B'01')**

GR affinity is not application-owned.

**APPL (B'10')**

GR affinity is application-owned.

**PRSISTVP**

Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**

The partner LU accepts persistent-verification indicators.

**NO (B'0')**

The partner LU does not accept persistent-verification indicators.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.
SENSE
The field in the RPL extension that returns a 32-bit sense code. This sense code is returned for the
control operator session that VTAM establishes as part of processing the CNOS request. This field is
labeled RPL6SNSI in the RPL extension.

USERFLD
Returns any unchanged user data that the application program placed in this field. This field is labeled
RPL6USR in the RPL extension.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
• VTAM-to-APPL required information vector (X'10')
• Partner's DCE capabilities vector (X'12')
• Name change vector (X'18')
• Partner's application capabilities vector (X'1A')

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues
this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0001'</td>
<td>OK—AS_SPECIFIED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0002'</td>
<td>OK—AS_NEGOTIATED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000C'</td>
<td>OK_AS_SPECIFIED—PARTNER_LU_KNOWN_BY_DIFFERENT_NAME</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000D'</td>
<td>OK_AS_NEGOTIATED—PARTNER_LU_KNOWN_BY_DIFFERENT_NAME</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0004'</td>
<td>OK—PARTNER LU SUPPORTS SINGLE SESSION</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0000'</td>
<td>CNOS_ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0001'</td>
<td>CNOS_ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0002'</td>
<td>CNOS_ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0003'</td>
<td>CNOS_ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0004'</td>
<td>CNOS_ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0005'</td>
<td>CNOS_ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0006'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0007'</td>
<td>NETWORK-QUALIFIED_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'000C'</td>
<td>X'0000'</td>
<td>CNOSRESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0000'</td>
<td>COMMAND_RACE_REJECT—PARTNER_GRANTED_RETRY</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0001'</td>
<td>COMMAND_RACE_REJECT—COPR_FOR_LOCAL_LU_TRIED</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0002'</td>
<td>COMMAND_RACE_REJECT—PARTNER_CNOS_IN_PROGRESS</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0003'</td>
<td>COMMAND_RACE_REJECT—LU_IS_IN_PENDING_SINGLE_STATE</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0004'</td>
<td>COMMAND_RACE_REJECT—PARTNER_LU_STARTING_SESSION</td>
</tr>
<tr>
<td>X'0020'</td>
<td>X'0000'</td>
<td>CNOS_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0028'</td>
<td>X'0000'</td>
<td>LU_MODE_SESSION_LIMIT_CLOSED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0004'</td>
<td>PARAMETER_ERROR—INVALID_VALUES_FOR_SNASVCMG_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0005'</td>
<td>PARAMETER_ERROR—INVALID_DRAINL_CHANGE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0006'</td>
<td>PARAMETER_ERROR—SNASVCMG_MODE_CANNOT_CURRENTLY_BE_RESET</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0007'</td>
<td>PARAMETER_ERROR—MINWINL_PLUS_MINWINR_EXCEEDS_SESSLIM</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0009'</td>
<td>PARAMETER_ERROR—INCOMPLETE_STRUCTURE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0018'</td>
<td>PARAMETER_ERROR—INVALID_LIMIT_SPECIFIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0019'</td>
<td>PARAMETER_ERROR—SNASVCMG_MODE_ALREADY_INITIALIZED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001A'</td>
<td>PARAMETER_ERROR—ALL_MODES_SPECIFIED_ON_SINGLE_SESSION_LU</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001B'</td>
<td>PARAMETER_ERROR—SNASVCMG_OR_CPSVCMG_MODE_FOR_SINGLE_SESSION_LU</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001C'</td>
<td>PARAMETER_ERROR—SINGLE_SESSION_MODE_ALREADY_INITIALIZED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001E'</td>
<td>CID_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0054'</td>
<td>X'0000'</td>
<td>UNRECOGNIZED_MODE_NAME</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=OPRCNTL, QUALIFY=DACTSESS**

**Purpose**
This macroinstruction responds negatively to a request for session establishment.

**Usage**
This command is issued after the application program is notified through its LOGON or SCIP exit routine that a CINIT or BIND request has been received. The function of this command is similar to the VTAM API commands CLSDST RELEASE and SESSIONC CONTROL=BIND for non-LU 6.2 sessions.

When this macroinstruction is used in a LOGON exit, the RPLAREA field of the read-only RPL contains a read-only copy of the CINIT. After examining the BIND image in the CINIT, the application program can issue this macroinstruction to prevent the session from being activated.

When this macroinstruction is used in a SCIP exit, the RPLAREA field of the read-only RPL contains the address of a read-only copy of the BIND. After examining the BIND, the application program can issue this macroinstruction to prevent the session from being activated.

APPCCMD CONTROL=OPRCNTL, QUALIFY=DACTSESS does not correspond to the DEACTIVATE_SESSION verb described in the LU 6.2 architecture.

**Context**
Input states are not applicable to this macroinstruction.

**Syntax**

```
name

APPCCMD CONTROL=OPRCNTL, QUALIFY=DACTSESS

, RPL = rpl_address_field
    ( rpl_address_register )

, AAREA = rpl_extension_address_field
    ( rpl_extension_address_register )
```
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4 ECB is meaningful only for asynchronous operations.

5 EXIT is meaningful only for asynchronous operations.

6 You can code more than one suboperand on OPTCD, but no more than one from each group.

7 KEEPSRB is meaningful only for synchronous operations.

8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:
AAREA=rpl_extension_address_field
AAREA= (rpl_extension_address_register)
    Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB= (acb_address_register)
    Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

ARG=4-byte_session_identifier_(cid)_field
ARG= (4-byte_session_identifier_(cid)_register)
    Specifies the CID of the session that was returned to the application program in the parameter list of the LOGON or SCIP exit routine. The specified CID must identify a CINIT or BIND that is queued for this application program.

BRANCH
    Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

    BRANCH=NO
        Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

    BRANCH=YES
        Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB
    Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

    ECB=INTERNAL
        Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

    ECB=ecb_address_field
    ECB= (ecb_address_register)
        Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT= (exit_routine_address_register)
    Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
    Specifies the following processing options that can be selected for the macroinstruction request:

    OPTCD=SYN
        Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

    OPTCD=ASY
        Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of
the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application
under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field
of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the
application under the same SRB in which VTAM was invoked. The indicator resides within the
RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the
processing of the APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of information that the application program can associate with this operator control
request. The information is returned unchanged when the macroinstruction completes. This data
cannot be used by any conversations. It can be used for correlation purposes. This field is labeled
RPL6USR in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application
program. It is labeled RPLFDB2 in the RPL.

RCPR
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the
application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is
labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the
application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is
labeled RPL6RCSC in the RPL extension. The combination of the RCPR and RCSEC fields indicates the
result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application
program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Returns any unchanged user data that the application program placed in this field. This field is labeled
RPL6USR in the RPL extension.

Return codes
The following (RCPR, RCSEC) combinations can be returned to the application program when it issues
this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE

Purpose

This macroinstruction changes the session limit values that have been defined and that are used to negotiate a CNOS request from a partner LU. It also displays selected fields from the LU-mode table in the DEFINE/DISPLAY (ISTSLD) control block.

Usage

This macroinstruction can be used to modify values in a mode name entry that were originally obtained by VTAM from the APPL definition statement or that were supplied by using this macroinstruction previously. There is no direct correlation to the DEFINE verb in the LU 6.2 architecture.

The session limit values that are defined are passed to VTAM in a DEFINE/DISPLAY control block. You must specify the address of this control block in the RPL when issuing the macroinstruction. The address is contained in the RPLAREA field, which can be set with the AREA keyword.

Most of the values specified in the DEFINE/DISPLAY control block are used to negotiate the values received in a CNOS request sent by the partner application program. The values are not affected by, nor do they have any effect upon, the values specified through the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction. For example, an application program can define the session limit used for negotiating purposes to be 10, yet later issue a CNOS macroinstruction that specifies a session limit of 20. The defined value of 10 does not restrict the CNOS value of 20; the CNOS value of 20 does not cause the defined value of 10 to be changed.

When this macroinstruction is issued before a CNOS request is negotiated on a mode, VTAM creates an entry in the LU-mode table for the mode and places the defined session limits in the table. The negotiated session limits are not determined until a CNOS request is negotiated.

When issuing APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS, the application program can elect not to specify the limits to be used for CNOS negotiation. If this occurs, VTAM uses the defined limits specified by this macroinstruction as the default for these values. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

This macroinstruction can also be used to help control VTAM's use of storage. Specifying default limits of 0 with DELETE=ALLOW (in the DEFINE/DISPLAY session limits control block) informs VTAM that this mode name can be deleted from the LU-mode table when the mode name is no longer being used. (Refer
to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for an example of setting the DEFINE/ DISPLAY session limits control block.) The execution of this macroinstruction involves only the application program; it does not cause any information to be sent through the network. The specified field values are in effect once the execution completes.

**Context**

Input states are not applicable to this macroinstruction.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```
APPCCMD      CONTROL   = OPRCNTL          QUALIFY   =

DEFINE 1

RPL       = rpl_address_field
           ( rpl_address_register )

AAREA        = rpl_extension_address_field
               ( rpl_extension_address_register )

ACB        = acb_address_field
               ( acb_address_register )

AREA        = define/display_session_limits_structure_address_field
               ( define/display_session_limits_structure_address_register )

BRANCH      = NO  YES

ECB        = INTERNAL
               ecb_address_field
               ( ecb_address_register )

EXIT        = exit_routine_address_field
               ( exit_routine_address_register )

LOGMODE      = 8-byte_logon_mode_name

LUNAME      = 8-byte.lu_name
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA** = *rpl_extension_address_field*

**AAREA** = *(rpl_extension_address_register)*

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB** = *acb_address_field*

**ACB** = *(acb_address_register)*

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

**AREA** = *define/display_session_limits_structure_address_field*

**AREA** = *(define/display_session_limits_structure_address_register)*

Specifies the address of a data area containing a DEFINE/DISPLAY session limits data structure. (See “DEFINE/DISPLAY session limits data structure (ISTSLD)” on page 587 for a description of the IBM-
supplied DSECT that can be used to map this storage.) A description of the fields in the control block can be found in z/OS Communications Server: SNA Programmer’s LU 6.2 Guide. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE=8-byte_logon_mode_name**

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

**LUNAME=8-byte.lu_name**

Specifies the name of the partner application program to which the change in the session limit and contention-winner polarity values applies. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. This field is labeled RPL6LU in the RPL extension.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program to which the change in the session limit and contention-winner polarity value applies.
If PARMS= (NQNames=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNames=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNames=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.) The network identifier also is used to find and update the contents of the logon mode table.

This network identifier is an identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=define/display_session_limits_structure_length**

**RECLEN=(define/display_session_limits_structure_register)**

Specifies the length of the DEFINE/DISPLAY session limits data structure supplied by the AREA field. The application program must supply the entire session limits data structure; it cannot supply a partial structure. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**USERFLD=4_bytes_of_user_data**

**USERFLD=(user_data_register)**

Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.
**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**
Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'000C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_BU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'0007'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—MINWINL_PLUS_MINWINR_EXCEEDS_SESSLIM</td>
</tr>
<tr>
<td>X'0009'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INCOMPLETE_STRUCTURE_SUPPLIED</td>
</tr>
<tr>
<td>X'000C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'000D'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'000E'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'000F'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'0017'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_MODE_SPECIFIED</td>
</tr>
<tr>
<td>X'001A'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002B'</td>
<td>X'0003'</td>
<td>NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0083'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABILE</td>
</tr>
<tr>
<td>X'0083'</td>
<td>X'0000'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0083'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'0083'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'0083'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'0083'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY

Purpose
This macroinstruction returns information associated with an LU or a mode name of an LU.

Usage
The information returned from this macroinstruction is contained in the DEFINE/DISPLAY control block. You must specify the address of this control block in the RPL when issuing the macroinstruction. It is contained in the RPLAREA field, which can be set with the AREA keyword. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of the control block.

The execution of this macroinstruction involves only the application program. It does not cause any information to be sent through the network. There is no direct correlation to the DISPLAY verb described in the LU 6.2 architecture.

Context
Conversation states are not applicable to this macroinstruction.

Syntax

```plaintext
APPCCMD name

CONTROL = OPRCNTL 

QUALIFY = DISPLAY

RPL = rpl_address_field

(AREA = rpl_address_register)

ACB = acb_address_field

(AREA = acb_address_register)

AREA = define/display_session_limits_structure_address_field

(AREA = define/display_session_limits_structure_address_register)

AREALEN = define/display_session_limits_structure_length

(AREA = define/display_session_limits_structure_length_register)
```

156 z/OS Communications Server: SNA Programmer's LU 6.2 Reference
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
Input parameters

The following information shows descriptions of the input parameters:

**AAREA** = *rpl_extension_address_field*

**AAREA** = *(rpl_extension_address_register)*

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB** = *acb_address_field*

**ACB** = *(acb_address_register)*

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA** = *define/display_session_limits_structure_address_field*

**AREA** = *(define/display_session_limits_structure_address_register)*

Specifies the address of a data area for the DEFINE/DISPLAY session limits data structure (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of this control block.). This field is labeled RPLAREA in the RPL.

**AREALEN** = *define/display_session_limits_structure_length*

**AREALEN** = *(define/display_session_limits_structure_length_register)*

Specifies the length of the area in which the DEFINE/DISPLAY session limits data structure is to be returned. If a mode name is specified for the LOGMODE field, the application program must supply an area large enough to contain the entire session limits data structure. If LOGMODE=0 is specified, a length of 40 can be coded for this field. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.
EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUNAME=8-byte_lu_name
Specifies the name of the partner application program to which the requested session information applies. The LU name is a name that is valid as the LU name value on the APPCCMD CONTROL=ALLOC macroinstruction and the network name of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program to which the requested session information applies.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.) The network identifier also is used to find and update the contents of the logon mode table. It is the same as the NETID value on the APPCCMD CONTROL=ALLOC macroinstruction.

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPL optional field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPL optional field of the RPL.
OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

USERFLD=4_byt es_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:

• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.
This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECL EN
The field in the RPL that returns to the application program the actual length of the session limits structure being returned by the AREA field. If the application program specified LOGMODE=0, the value 40 is returned for this field. This field is labeled RPLRELEN in the RPL.
A description of the session limits structure is found in the z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**
Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X’10’)
- Partner’s DCE capabilities vector (X’12’)
- Partner’s application capabilities vector (X’1A’)

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS.OR.LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0013'</td>
<td>PARAMETER_ERROR—NO_CORRESPONDING_MODE_IN_LOGMODE_TABLE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0016'</td>
<td>PARAMETER_ERROR—NO_CORRESPONDING_LU_IN_LOGMODE_TABLE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'00000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE

Purpose
This macroinstruction is used to restore modes and their associated persistent LU-LU sessions that are pending recovery.

Usage
This macroinstruction can be used by an application program to restore modes and associated persistent LU-LU sessions that are pending recovery. A mode is restored only after any sessions for the mode are restored. A mode without sessions also must be restored.

A single LU-mode can be restored when the LU name and logon mode are specified on the RESTORE command. All modes for a specific LU are restored when only the LU name is specified. If neither the LU name (with its NETID, if applicable) nor the logon mode is specified, all LUs and modes in the LU-mode table are restored.

The application program specifies the amount of information that is to be returned in the RESTORE control block. To do this, it uses the LIST keyword in the RESTORE macroinstruction. The application program can specify LU-mode table information, LU-mode table and session information, or no information. If the application program requests information to be returned, it must specify the address of a data area to contain that information. The application program must provide the storage area in addition to specifying the address of the storage. This address is contained in the RPLAREA field, which can be set with the AREA keyword.

When the area pointed to by RPLAREA is large enough, the macroinstruction builds multiple RESTORE blocks in this area, if necessary. The RESTORE structures are placed in the area specified until the area is filled or the command is completed, whichever comes first.

For more information about the restore process, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide. For information about the RESTORE control block, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide. For an example of retrieving information that is returned, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context
Input states are not applicable to this macroinstruction.

The recovering VTAM application program can issue this macroinstruction only after it issues the SETLOGON START macroinstruction. Otherwise, this macroinstruction is rejected.

Syntax

```plaintext
APPCCMD  CONTROL=OPRCNTL  QUALIFY=RESTORE

name

RPL = rpl_address_field
    ( rpl_address_register )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=restore_structure_address_field**

**AREA=(restore_structure_address_register)**

Specifies the address of a data area that returns one or more RESTORE data structures. It is used only with LIST=ALL or LIST=NOSESS. It is unnecessary when LIST=NONE is specified. This field is labeled RPLAREA in the RPL.

**AREALEN=restore_structure_length**

**AREALEN=(restore_structure_length_register)**

Specifies the length of the area in which the RESTORE data structure is to be returned. It is used only with LIST=ALL or LIST=NOSESS. It is unnecessary when LIST=NONE is specified. This field is labeled RPLBUFL in the RPL.
BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LIST
Specifies the information to be returned in the RESTORE structure, which describes the LUs, modes, and sessions that have been restored. This field is labeled RPL6LIST in the RPL extension.

LIST=ALL
Specifies that all LU, mode, and session information is returned in the RESTORE structure.

LIST=NONE
Specifies that no RESTORE structure is returned.

LIST=NOSESS
Specifies that all LU and mode information but no session information is returned in the RESTORE structure.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name which should be restored. The application program can specify a logon mode name with an LU name to give greater granularity over the scope of the command. LOGMODE can be specified only with LUNAME. The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is less than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6MODE in the RPL extension.

LUNAME=8-byte_lu_name
Specifies the name of the partner whose modes must be restored. It is the same as the LU name value on the APPCCMD CONTROL=ALLOC macroinstruction. It is also the network name of the target LU. When the application program does not specify the LU name, all LUs and modes are restored. Otherwise, only the modes associated with a specified LU name are restored. The LU name can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. This field is labeled RPL6LU in the RPL extension.
NETID=8-byte_network_identifier
Specifies the network identifier of the partner whose modes must be restored.

If PARMS=(NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored.

If NQNAME=YES is specified, LUNAME and NETID together form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.) The network identifier also is used to find the appropriate information on sessions and modes waiting to be restored. It is the same as the NETID value on the APPCCMD CONTROL=ALLOC macroinstruction.

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPR
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.
RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
The field in the RPL that returns to the application program the length of AREA used to contain the RESTORE structure(s) returned by the AREA field. This field is labeled RPLRLEN in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (RESTORE complete.)</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0006'</td>
<td>RESTORE_UNNECESSARY—NO_SESSIONS_TO_RESTORE</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0007'</td>
<td>RESTORE_INCOMPLETE—INPUT_WORK_AREA_TOO_SMALL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0013'</td>
<td>NO_CORRESPONDING_MODE_IN_LM_TABLE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0016'</td>
<td>NO_CORRESPONDING_LU_IN_LM_TABLE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0029'</td>
<td>INVALID_LIST_VALUE_SPECIFIED_ON_APPCCMD_FOR_RESTORE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'007B'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'009C'</td>
<td>X'0001'</td>
<td>RESTORE_REJECTED—RESTORE_ISSUED_BEFORE_SETLOGON_START</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=PREALLOC, QUALIFY=ALLOCD

Purpose
This macroinstruction reserves a session without establishing a conversation. If a session is not available and session limits allow, VTAM activates a session for the conversation, if possible. Session related information can be passed from VTAM to the application before the application sends the FMH-5. The conversation is not active until the application issues the APPCCMD CONTROL=SENDFMH5 macroinstruction.

Usage
QUALIFY=ALLOCD is used when an application program preallocates a conversation and wants VTAM to queue the request if the request cannot be met immediately. This macroinstruction completes when VTAM reserves a session for a conversation or when an error occurs that prevents VTAM from reserving a session.

VTAM finds a session for the conversation as follows:
1. If a session is free, VTAM reserves it for a conversation.
2. If no free sessions exist and session limits allow, VTAM establishes a session and reserves it for a conversation.
3. If a new session cannot be established, VTAM queues the request until a session becomes available or until the session limits are changed to allow the establishment of a new session.

After session initiation, the conversation is reserved in PENDING ALLOCATE state and the application receives the conversation identifier in the CONVID field. The application could also receive the PCID for the session if VTRINA and VTRINL are specified on the preallocation request. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.

Context
This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is suspended for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax

```
name APPCCMD CONTROL = PREALLOC , QUALIFY = ALLOCD
RPL = rpl_address_field ( rpl_address_register )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=**rpl_extension_address_field
**AAREA=(**rpl_extension_address_register**)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**acb_address_field
**ACB=(**acb_address_register**)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with
transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**
  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**
  
  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- **CONMODE=BUFFCA**
  
  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

- **CONMODE=CS**
  
  Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

- **CONMODE=LLCA**
  
  Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

- **CONXMOD=CA**
  
  Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

- **CONXMOD=CS**
  
  Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPL6OPT1 field of the RPL.

- **ECB=INTERNAL**
  
  Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.
ECB=ecb_address_field
ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name

Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

LUAFFIN=APPL

The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

LUNAME=8-byte_lu_name

Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NAMEUSE

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

NAMEUSE=APNAME

The application identifies itself to the partner LU by its application network name.
NAMEUSE=GNAME
The application identifies itself to the partner LU by a generic resource name.
The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)
This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.
USERFLD=4-bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:

• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.
This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B'1')
The partner LU accepts the already-verified indicator.

NO (B'0')
The partner LU does not accept the already-verified indicator.

CGID
Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

CONSTATE
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.
This field can have the following values at the completion of this macroinstruction:

X'00'
RESET
X'08'
END_CONV
X'FF'
PENDING_ALLOCATE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.
Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

CONVSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is returned at completion only if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.
PRSISTVP
 Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**
The partner LU accepts persistent-verification indicators.

**NO (B'0')**
The partner LU does not accept persistent-verification indicators.

RCPRI
 The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
 The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
 The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
 The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

SESSID
 The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
 The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
 The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**
The session was established using session-level LU-LU verification.

**NO (B'0')**
The session was not established using session-level LU-LU verification.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X'10')
- Partner's DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner's nonce vector (X'14')
- Send FMH_5 sequence number vector (X'15')
- Receive FMH_5 sequence number vector (X'16')
• PCID vector (X'17')
• Name change vector (X'18')
• Session information vector (X'19')
• Partner's application capabilities vector (X'1A')

**State changes**

The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>PARAMETER_ERROR—NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=PREALLOC, QUALIFY=CONVGRP

#### Purpose
This macroinstruction reserves a session for a conversation with a specified conversation group identifier without establishing a conversation. If the specified session is not available and session limits allow, VTAM queues the request until the session becomes available. If the specific session does not exist, VTAM fails the preallocation request. After a session is reserved, session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDFMH5 is issued.

#### Usage
QUALIFY=CONVGRP is used to preallocate a conversation over a specific session that already exists. It provides the ability to serially preallocate a related group of conversations on a particular session. This macroinstruction completes when:

- VTAM assigns the specified session to the conversation.
- The specific session is deactivated.
- An error occurs that prevents VTAM from assigning the session to the conversation.

To indicate the session to be used, the application program specifies the conversation group identifier for the session on the CGID keyword. The conversation group identifier of the session is returned to the application program by the CGID returned field for the following APPCCMD macroinstructions:

- APPCCMD CONTROL=ALLOC
- APPCCMD CONTROL=PREALLOC
- APPCCMD CONTROL=RCVFMH5

VTAM finds the session for the conversation as follows:

1. If the specified session is available, VTAM assigns it to the conversation.
2. If the specified session exists but is not available, VTAM queues the request until the session becomes available.
3. If the specified session is deactivated while the request is queued, the queued request is rejected.

After session initiation, the session is reserved to receive session related information if necessary and is assigned to a conversation. A conversation identifier is returned to the application in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).
The application program can specify how expedited data is to be received on this conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```plaintext
name APPCCMD CONTROL PREALLOC QUALIFY

CONVGRP

RPL rpl_address_field

AAREA rpl_extension_address_field

ACB acb_address_field

BRANCH NO YES

CGID 32-bit_conversation_group_id_field

CONMODE BUFFCA CS LLCA

CONXMOD CA CS
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
Input parameters

The following information shows descriptions of the input parameters:

AAREA=\texttt{rpl\_extension\_address\_field}
AAREA=\texttt{(rpl\_extension\_address\_register)}
\textbf{Specifies} the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\texttt{acb\_address\_field}
ACB=\texttt{(acb\_address\_register)}
\textbf{Specifies} the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
\textbf{Specifies} whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
\textbf{Authorized path processing is not to be used.} For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
\textbf{Authorized path processing is to be used.} For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CGID=\texttt{32\text{-}bit\_conversation\_group\_id\_field}
CGID=\texttt{(32\text{-}bit\_conversation\_group\_id\_register)}
\textbf{Specifies} the 32-bit conversation group ID.
\textbf{This value can} be obtained from a previous APPCCMD CONTROL=ALLOC, CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction. If the CGID operand is not specified, VTAM uses the conversation group ID that is already in the RPL6CGID field on the RPL extension.

The conversation group ID identifies a specific session between two specific LUs. It provides a means by which a VTAM LU 6.2 application program and its partner LU can share serially the same session.

CONMODE
\textbf{Specifies} the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

\begin{itemize}
\item \textbf{CONMODE=BUFFCA}
\textbf{Specifies} that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

\item \textbf{CONMODE=CS}
\textbf{Specifies} that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the subsequent APPCCMD CONTROL=SENDFH5. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, for example, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, for example, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event control block address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.
**NAMEUSE**
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**
The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**
The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**
**RPL=(rpl_address_register)**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

**RTSRTRN=EXPD**
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

**USERFLD=4-bytes_of_user_data**
**USERFLD=(user_data_register)**
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.
VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:
• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.
This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B'1')
The partner LU accepts the already-verified indicator.

NO (B'0')
The partner LU does not accept the already-verified indicator.

CONSTATE
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.
This field can have the following values at the completion of this macroinstruction:

X'00'
RESET

X'08'
END_CONV

X'FF'
PENDING_ALLOCATE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

CONVSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.
NO (B'0')
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGMODE
Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is returned at completion only if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.
NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

LUNAME
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NETID
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

SESSID
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

SESSLIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.
**YES (B'1')**
The session was established using session-level LU-LU verification.

**NO (B'0')**
The session was not established using session-level LU-LU verification.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- Partner's DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner's nonce vector (X'14')
- Send FMH_5 sequence number vector (X'15')
- Receive FMH_5 sequence number vector (X'16')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

**State changes**
The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APB_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APB_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002A'</td>
<td>PARAMETER_ERROR—INVALID_CGID_VALUE_SPECIFIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_ORRESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_REALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=PREALLOC, QUALIFY=CONWIN**

**Purpose**

This macroinstruction reserves a contention-winner session for a conversation, if session limits allow, without establishing a conversation. If a contention-winner session is not available, VTAM queues the request for later completion. After a session is reserved, Session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDMH5 is issued.

**Usage**

QUALIFY=CONWIN is used when an application program preallocates a conversation and wants VTAM to queue the request if no contention-winner session can be assigned. This macroinstruction completes when VTAM reserves a contention-winner session or an error occurs that prevents VTAM from assigning a session.

VTAM finds a session for the conversation as follows:

1. If a contention-winner session is currently available, VTAM reserves it for a conversation.
2. If no contention-winner session is available and session limits allow, VTAM establishes a new contention-winner session and assigns it to the conversation.
3. If a new contention-winner session cannot be established, VTAM queues the request until a contention-winner session is available or session limits are changed to allow a new contention-winner session to be activated.

For this macroinstruction to complete successfully, the session limits must define at least one contention-winner session.

If contention-winner sessions are deactivated under normal conditions and an APPCCMD CONTROL=PREALLOC, QUALIFY=CONWIN request is queued, VTAM activates another contention-winner session to meet the queued request.

After session initiation, the session is reserved to receive session related information if necessary and is assigned to a conversation. A conversation identifier is returned to the application in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.
Context
This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.
When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax

```plaintext
name

APPCCMD — CONTROL — PREALLOC —, QUALIFY — = — CONWIN

RPL = rpl_address_field

( rpl_address_register )

AAREA = rpl_extension_address_field

( rpl_extension_address_register )

ACB = acb_address_field

( acb_address_register )

BRANCH = NO

YES

CONMODE = BUFFCA

CS

LLCA

CONXMOD = CA

CS

ECB = INTERNAL

( ecb_address_field )

( ecb_address_register )

EXIT = exit_routine_address_field

( exit_routine_address_register )

LOGMODE = 8-byte_logon_mode_name
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

- **AAREA=**\(rpl\_extension\_address\_field\)
  - **AAREA=(rpl\_extension\_address\_register)**
    - Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

- **ACB=**\(acb\_address\_field\)
  - **ACB=(acb\_address\_register)**
    - Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

- Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
  - **BRANCH=NO**
    - Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.
  - **BRANCH=YES**
    - Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

- Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.
  - **CONMODE=BUFCA**
    - Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.
  - **CONMODE=CS**
    - Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.
  - **CONMODE=LLCA**
    - Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

- Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.
CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|;IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN
Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

LUAFFIN=APPL
The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL
VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU,
the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**LUNAME=8-byte_lu_name**
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**
The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**
The application identifies itself to the partner LU by a generic resource name.
The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.
RPL=rpl_address_field
RPL=(rpl_address_register)
    Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
    Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

    RTSRTRN=BOTH
        Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

    RTSRTRN=EXPD
        Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4-bytes_of_user_data
USERFLD=(user_data_register)
    Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
    Specifies the address of the data area where VTAM places vector list information for the application.

    This parameter is ignored if one of the following items is true:
    • VTRINA=0
    • The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
    • The value for VTRINL is not specified.

    This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
    Specifies the length of the data area where VTAM places vector list information for the application.

    This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

    AVFA
        The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

        YES (B'1')
            The partner LU accepts the already-verified indicator.

        NO (B'0')
            The partner LU does not accept the already-verified indicator.

    CGID
        Specifies the 32-bit conversation group identifier.

        It is labeled RPL6CGID in the RPL extension.
CONSTATE
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.

This field can have the following values at the completion of this macroinstruction:

X’00’
  RESET
X’08’
  END_CONV
X’FF’
  PENDING_ALLOCATE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

CONVERSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B’1’)
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B’0’)
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B’00’)
  No data is to be encrypted.

SELECTIVE (B’01’)
  The application program specifies the data that is to be encrypted.

REQUIRED (B’11’)
  All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B’1’)
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program.
program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**LUAFFIN**
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

**NONE (B'00')**
GR affinity is not applicable or is unknown.

**NOTAPPL (B'01')**
GR affinity is not application-owned.

**APPL (B'10')**
GR affinity is application-owned.

**PRSISTVP**
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**
The partner LU accepts persistent-verification indicators.

**NO (B'0')**
The partner LU does not accept persistent-verification indicators.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.
YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
• VTAM-to-APPL required information vector (X'10')
• Partner's DCE capabilities vector (X'12')
• Local nonce vector (X'13')
• Partner's nonce vector (X'14')
• Send FMH_5 sequence number vector (X'15')
• Receive FMH_5 sequence number vector (X'16')
• PCID vector (X'17')
• Name change vector (X'18')
• Session information vector (X'19')
• Partner's application capabilities vector (X'1A')

State changes
The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>PARAMETER_ERROR—NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=PREALLOC, QUALIFY=IMMED

#### Purpose
This macroinstruction reserves an active contention-winner session for a conversation, if session limits allow, without establishing a conversation. If no session is available, the preallocation request fails. After a session is reserved, session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDFMH5 is issued.

#### Usage
QUALIFY=IMMED is used to preallocate a conversation when the application program needs an immediate response from VTAM. This macroinstruction completes successfully only when an active contention-winner session is available to be reserved for a conversation. If the request cannot be met immediately, VTAM does not queue it. VTAM neither tries to activate a new session nor bids on a contention-loser session.

When a conversation is preallocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program must associate a conversation with a particular transaction by using the conversation identifier.

The application program can specify how expedited data is to be received.
Context
This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax

```
name APPCCMD CONTROL = PREALLOC , QUALIFY = IMMED

, RPL = rpl_address_field ( rpl_address_register )

, AAREA = rpl_extension_address_field ( rpl_extension_address_register )

, ACB = acb_address_field ( acb_address_register )

, BRANCH = NO YES

, CONMODE = BUFFCA CS LLCA

, CONXMOD = CA CS

, ECB = ecb_address_field ( ecb_address_register )

, EXIT = exit_routine_address_field ( exit_routine_address_register )

, LOGMODE = 8-byte_logon_mode_name
```
LUNAME = 8-byte_lu_name

NETID = 8-byte_network_identifier

OPTCD = (ASY 3, SYN 3, KEEPSRB 3 7, NKEEPSRB 3 8)

RTSRTRN = BOTH 1

USERFLD = 4-bytes_of_user_data

VTRINA = vector_address_field

VTRINL = vector_length_field

Notes:
1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters
The following information shows descriptions of the input parameters:
**AAREA=** rpl_extension_address_field

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** acb_address_field

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.
CONXMOD=CS
  Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD
  CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
  Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single
  APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

  ECB=INTERNAL
    Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

  ECB=ecb_address_field
    Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD
    completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be
    any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
  Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD
  completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The
  indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
  Specifies the logon mode name designating the network properties for the session to be preallocated
  for this conversation. The network properties include, for example, the class of service to be used.

  The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If
  it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

  If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that
does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default
  mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA
  Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon
  mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build
  the logon mode table named in the MODETAB operand of the APPL definition statement for this
  application program.) For more information on the MODEENT macroinstruction, refer to z/OS
  Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the
  RPL extension.

LUNAME=8-byte.lu_name
  Specifies the name of the partner application program at which the remote transaction program,
specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of
  the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the
  LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier
  Specifies the network identifier of the partner application program at which the remote transaction
  program, specified in the FMH-5 supplied in the AREA field, is found.

  If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the
  NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID
  must be supplied.

  If NQNAMES=Yes, LUNAME and NETID are used together to form the network-qualified name of the
  target LU. (If NETID is specified, LUNAME must be specified.)

  This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is
  fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is
  labeled RPL6NET in the RPL extension.

OPTCD
  Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4-bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application.
This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the RPL and RPL extension fields:

**AVFA**
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

- **YES (B'1')**
  The partner LU accepts the already-verified indicator.

- **NO (B'0')**
  The partner LU does not accept the already-verified indicator.

**CGID**
Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

**CONSTATE**
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.
This field can have the following value at the completion of this macroinstruction:

- **X'00'**
  RESET

- **X'08'**
  END_CONV

- **X'FF'**
  PENDING_ALLOCATE

**CONVID**
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

*Note:* The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

**CONVSECP**
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

- **YES (B'1')**
  The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

- **NO (B'0')**
  The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

- **NONE (B'00')**
  No data is to be encrypted.

- **SELECTIVE (B'01')**
  The application program specifies the data that is to be encrypted.
**REQUIRED (B'11')**
All data is to be encrypted.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**
  No FMH-5s are waiting to be received by the application program.

**PRSISTVP**
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

- **YES (B'1')**
  The partner LU accepts persistent-verification indicators.

- **NO (B'0')**
  The partner LU does not accept persistent-verification indicators.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension. The format of the session instance identifier is described in the z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.
YES (B'1')
Indicates that the session was established using session-level LU-LU verification.

NO (B'0')
Indicates that the session was not established using session-level LU-LU verification.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- Partner's DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner's nonce vector (X'14')
- Send FMH_5 sequence number vector (X'15')
- Receive FMH_5 sequence number vector (X'16')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

State changes
The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>PARAMETER_ERROR—NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0058'</td>
<td>X'0000'</td>
<td>UNSUCCESSFUL_SESSION_NOT_AVAILABLE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
</tbody>
</table>
## APPCCMD CONTROL=PREALLOC, QUALIFY=WHENFREE

### Purpose
This macroinstruction reserves a session for a conversation, if session limits allow, without establishing a conversation. If a session is not available and one cannot be activated, VTAM returns control to the application program. After a session is reserved, session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDFMH5 is issued.

### Usage
QUALIFY=WHENFREE is used when an application program preallocates a conversation and wants VTAM to search for a session that satisfies the ALLOCATE request. This macroinstruction completes when VTAM reserves a session for a conversation or when VTAM cannot reserve a session and returns control to the application program with a return code of X'0004', X'0001'.

VTAM finds a session for the conversation as follows:

1. If a session is available, VTAM reserves it for a conversation.
2. If no available sessions exist and session limits allow, VTAM establishes a session and reserves it for a conversation.
3. If a session cannot be established and session activation requests are pending, VTAM queues the PREALLOCATE request until the request is satisfied or until all pending session activation requests are used. If the pending session activation requests are used before the PREALLOCATE request is satisfied, VTAM fails the PREALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001'.
4. If a session cannot be established and no session activation request is pending that might satisfy the PREALLOCATE request, VTAM fails the PREALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001' and returns control to the application program.
After session initiation, the session is reserved to receive session related information if necessary and is assigned to a conversation. When a conversation is preallocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```plaintext
APPCCMD   --- CONTROL   --- PREALLOC   --- QUALIFY   ---

WHENFREE 1,

RPL = rpl_address_field
     ( rpl_address_register )

AAREA = rpl_extension_address_field
        ( rpl_extension_address_register )

ACB = acb_address_field
     ( acb_address_register )

BRANCH = NO
         YES

CONMODE = BUFFCA
          CS
          LLCA

CONXMOD = CA
          CS
```
ECB = INTERNAL (ecb_address_field)

EXIT = exit_routine_address_field

LOGMODE = 8-byte_logon_mode_name

LUAFFIN = APPL

LUNAME = 8-byte_lu_name

NAMEUSE = APNAME

NETID = 8-byte_network_identifier

OPTCD = (ASY, SYN, KEEPSRB, NKEEPSRB)

RTSRTRN = BOTH

USERFLD = 4-bytes_of_user_data

VTRINA = vector_address_field
Notes:

1.Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** *(rpl_extension_address_field)*

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** *(acb_address_field)*

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.
CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS
Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**LUNAME=8-byte_lu_name**

Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**

The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**

The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
   Specifies that control is to be returned synchronously to the application program when the
   function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the
   RPL.

OPTCD=ASY
   Specifies that control is to be returned to the application program immediately and that the
   application program is to be informed later of the completion of the macroinstruction by the
   posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of
   the RPL.

OPTCD=KEEPSRB
   Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application
   under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field
   of the RPL.

OPTCD=NKEEPSRB
   Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the
   application under the same SRB in which VTAM was invoked. The indicator resides within the
   RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
   Specifies the address of the request parameter list that contains information to be used during the
   processing of the APPCCMD macroinstruction.

RTSRTRN
   Specifies the manner in which the Request_To_Send_Received indication is to be reported to the
   application on subsequent macroinstructions.

   RTSRTRN=BOTH
      Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and
      SIGDATA fields on all APPCCMDs that return these parameters.

   RTSRTRN=EXPD
      Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and
      SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4-bytes_of_user_data
USERFLD=(user_data_register)
   Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD
   macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL
   extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the
   conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5
   macroinstruction (if the conversation was initiated by a remote application program). This field is
   labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
   Specifies the address of the data area where VTAM places vector list information for the application.
   This parameter is ignored if one of the following items is true:
   - VTRINA=0
   - The value for VTRINL is less than the minimum length required to return the APPCCMD vector area
     header.
   - The value for VTRINL is not specified.
   This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
   Specifies the length of the data area where VTAM places vector list information for the application.
This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the RPL and RPL extension fields:

**AVFA**
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

**YES (B'1')**
The partner LU accepts the already-verified indicator.

**NO (B'0')**
The partner LU does not accept the already-verified indicator.

**CGID**
Specifies the 32-bit conversation group identifier.

It is labeled RPL6CGID in the RPL extension.

**CONSTATE**
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.

This field can have the following value at the completion of this macroinstruction:

- **X'00'**
  - RESET

- **X'08'**
  - END_CONV

- **X'FF'**
  - PENDING_ALLOCATE

**CONVID**
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

**CONVSECP**
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSAS in the RPL extension.

**YES (B'1')**
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

**NO (B'0')**
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

**NONE (B'00')**
No data is to be encrypted.

**SELECTIVE (B'01')**
The application program specifies the data that is to be encrypted.
REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMHSLEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.
SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

SESSID
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- Partner's DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner's nonce vector (X'14')
- Send FMH_5 sequence number vector (X'15')
- Receive FMH_5 sequence number vector (X'16')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

State changes
The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>PARAMETER_ERROR—NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE.</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE.</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=PREPRCV, QUALIFY=CONFIRM

**Purpose**
This macroinstruction is used to change the local conversation state of a half-duplex conversation from SEND to RECEIVE. This macroinstruction flushes the SEND buffer and then sends a confirmation request to the partner application program. When a positive acknowledgment to the confirmation is received, the macroinstruction changes the conversation state from SEND to RECEIVE.

**Usage**
This macroinstruction synchronizes the communication between the local and remote LUs. It is issued when the application program has finished sending and is ready to receive. This macroinstruction causes VTAM to flush the SEND buffer (in the same way as it does for APPCCMD CONTROL=SEND, QUALIFY=CONFIRM) and send a confirmation request to the partner LU.

If a positive acknowledgment to the confirmation is received (as indicated by an RCPRI of X’0000’), VTAM changes the conversation from SEND to RECEIVE state in preparation to receive data. If a negative confirmation response is received (RCPRI is not X’0000’), the state of the conversation is found in the CONSTATE field.

This macroinstruction corresponds to the PREPARE_TO_RECEIVE (TYPE=CONFIRM) verb described in the LU 6.2 architecture.

**Context**
This macroinstruction can be issued only from SEND or PENDING_SEND conversation state. This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

The local application can specify whether this acknowledgment is a response (LOCKS=SHORT) or data received from the partner (LOCKS=LONG). The LOCKS=SHORT specification completes more quickly and the LOCKS=LONG specification uses fewer transmission flows and processing cycles.

**Syntax**

```
name APPCCMD  —  CONTROL  =  PREPRCV  ,  QUALIFY  =  CONFIRM

,  RPL  =  rpl_address_field
   (  rpl_address_register  )

,  AAREA  =  rpl_extension_address_field
   (  rpl_extension_address_register  )

,  ACB  =  acb_address_field
   (  acb_address_register  )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=**<ruby>assertion_extension_address_field</ruby>
**AAREA=**(assertion_extension_address_register)

- Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**<ruby>acb_address_field</ruby>
**ACB=**(acb_address_register)

- Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

- Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

  - BRANCH=NO
    - Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

  - BRANCH=YES
    - Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

- Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

  - CONMODE=BUFFCA
    - Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

  - CONMODE=CS
    - Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOCKS
Specifies when the execution of the macroinstruction is complete following execution of the CONFIRM function. This field corresponds to the LOCKS parameter on the PREPARE_TO_RECEIVE verb, as described in the LU 6.2 architecture. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information on the use of this function.) This field is labeled RPL6LOCK in the RPL extension.

LOCKS=LONG
Specifies that the function of this macroinstruction is complete when information, such as data, is received from the partner application. The receipt of data presumes an affirmative reply to the
confirmation request. The local application program must issue an APPCCMD CONTROL=RECEIVE in order to get the information that caused the macroinstruction to complete.

**LOCKS=SHORT**
Specifies that the function of this macroinstruction is complete when a positive response is received to the confirmation request.

**Note:** The partner cannot determine whether LOCKS=LONG or SHORT was specified. The APPCCMD CONTROL=SEND, QUALIFY=CONFRMD must be specified in either case.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**
**RPL=(rpl_address_register)**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**VTRINA=vector_address_field**
**VTRINA=(vector_address_register)**
Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

**VTRINL=vector_length_field**
**VTRINL=(vector_length_register)**
Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.
This field can have the following values:

- **X'01'**
  - SEND

- **X'02'**
  - RECEIVE

- **X'03'**
  - RECEIVE_CONFIRM

- **X'04'**
  - RECEIVE_CONFIRM_SEND

- **X'05'**
  - RECEIVE_CONFIRM DEALLOCATE

- **X'07'**
  - PENDING_END_CONVERSATION_LOG

- **X'08'**
  - END_CONVERSATION

- **X'09'**
  - PENDING_SEND

- **X'0A'**
  - PENDING_RECEIVE_LOG

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. It is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**LOGRCV**

The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**YES (B'1')**

An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:
ALLOCATION_ERROR  
DEALLOCATE_ABEND_PROGRAM  
DEALLOCATE_ABEND_SERVICE  
DEALLOCATE_ABEND_TIMER  
PROGRAM_ERROR_NO_TRUNC  
PROGRAM_ERROR_PURGING  
PROGRAM_ERROR_TRUNC  
SERVICE_ERROR_NO_TRUNC  
SERVICE_ERROR_PURGING  
SERVICE_ERROR_TRUNC  
USER_ERROR_CODE_RECEIVED  

NO (B'0')  
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI  
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC  
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD  
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE  
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM. This field is labeled RPL6SNSI in the RPL extension.

USERFLD  
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.
State changes

These changes are applicable when RCPRI indicates OK.
The conversation state is RECEIVE after successful processing.
See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPLE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—EXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=PREPRCV, QUALIFY=DATACON**

**Purpose**

This macroinstruction sends data, flushes the SEND buffer, and then sends a confirmation request to the partner application program. If a positive confirmation acknowledgment is received, the local conversation state is changed from SEND to RECEIVE state.

**Usage**

This macroinstruction combines the functions of two macroinstructions, APPCCMD CONTROL=SEND, QUALIFY=DATA followed by APPCCMD CONTROL=PREPRCV, QUALIFY=CONFIRM. VTAM flushes the SEND buffer and sends the data that is specified on the macroinstruction. A confirmation request follows. The application program must ensure that the data sent completes a logical record.

If a positive acknowledgment to the confirmation request is received, the conversation is placed in RECEIVE state. When this macroinstruction completes without error, the state of the conversation is contained in the CONSTATE field.

This macroinstruction corresponds to the verbs SEND_DATA followed by PREPARE_TO_RECEIVE (TYPE=CONFIRM) described in the LU 6.2 architecture.

**Context**

This macroinstruction can be issued from the SEND or PENDING_SEND conversation state. This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```plaintext
APPCCMD  CONTROL = PREPRCV  ,  QUALIFY = DATACON
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

Following are descriptions of the input parameters:

**AAREA=** *rpl_extension_address_field*

**AAREA=(** *rpl_extension_address_register*)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** *acb_address_field*

**ACB=(** *acb_address_register*)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=** *data_area_or_buffer_list_address_field*

**AREA=(** *data_area_or_buffer_list_address_register*)

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this
conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=XBULST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBULST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.
CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOCKS
Specifies when the execution of the macroinstruction is complete following execution of the CONFIRM function. This field corresponds to the LOCKS parameter on the PREPARE_TO_RECEIVE verb, as described in the LU 6.2 architecture. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information on the use of this function.) This field is labeled RPL6LOCK in the RPL extension.

LOCKS=LONG
Specifies that the function of this macroinstruction is complete when information, such as data, is received from the partner application. The receipt of data presumes an affirmative reply to the
confirmation request. The local application program must issue an APPCCMD CONTROL=RECEIVE in order to get the information that caused the macroinstruction to complete.

LOCKS=SHORT
Specifies that the function of this macroinstruction is complete when a positive response is received to the confirmation request.

Note: The partner cannot determine whether LOCKS=LONG or SHORT was specified. The APPCCMD CONTROL=SEND, QUALIFY=CONFRMD must be specified in either case.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

OPTCD=XBUFFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFFLST.
• Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X'002C', X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

**RECLEN**=\(data\_length\)

**RECLEN**=(\(data\_length\_register\))

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

\*RPL*=\(rpl\_address\_field\)

\*RPL*=\(rpl\_address\_register\)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

Following are descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

This field can have the following values:

- X'01'
  SEND
- X'02'
  RECEIVE
- X'03'
  RECEIVE_CONFIRM
- X'04'
  RECEIVE_CONFIRM_SEND
- X'05'
  RECEIVE_CONFIRM_DEALLOCATE
- X'07'
  PENDING_END_CONVERSATION_LOG
- X'08'
  END_CONVERSATION
- X'09'
  PENDING_SEND
- X'0A'
  PENDING_RECEIVE_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.
**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. It is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**LOGRCV**

The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**YES (B'1')**

An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004'
  - ALLOCATION_ERROR
- X'0014'
  - DEALLOCATE_ABEND_PROGRAM
- X'0018'
  - DEALLOCATE_ABEND_SERVICE
- X'001C'
  - DEALLOCATE_ABEND_TIMER
- X'0030'
  - PROGRAM_ERROR_NO_TRUNC
- X'0034'
  - PROGRAM_ERROR_PURGING
- X'0038'
  - PROGRAM_ERROR_TRUNC
- X'003C'
  - SERVICE_ERROR_NO_TRUNC
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RPLXSRV**

A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application’s responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM. This field is labeled RPL6SNSI in the RPL extension.

**SIGDATA**

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. It is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**

The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.
**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVMH5), the application specified RTSRTN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). This field is labeled RPL6RSIG in the RPL extension.

**YES (B'1')**
A SIGNAL RU has been received from the partner application program. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner application program. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**STSHBF**
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
These changes are applicable when RCPRI indicates OK.

The conversation enters RECEIVE state after successful completion of the macroinstruction.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (REMOTE PROGRAM REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>RCRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED—CONTROL/QUALIFY_VALUE_NOT_VALID_FOR_FULL-DUPLEX_CONVERSATIONS</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=PREPRCV, QUALIFY=DATAFLU

Purpose
This macroinstruction sends data to a partner LU and flushes the SEND buffer. The conversation state for
the application program is then changed from SEND to RECEIVE.

Usage
This macroinstruction combines the functions of two macroinstructions, APPCCMD CONTROL=SEND,
QUALIFY=DATA followed by APPCCMD CONTROL=PREPRCV, QUALIFY=FLUSH. VTAM sends any data
currently in the SEND buffer. This data is followed by the data specified on the macroinstruction to the
partner LU. The application program must ensure that the data sent completes a logical record.

If the data is sent successfully, the conversation is placed in RECEIVE state. The conversation state is
found in the CONSTATE field when the macroinstruction completes.

This macroinstruction corresponds to the SEND_DATA followed by PREPARE_TO_RECEIVE (TYPE=FLUSH)
verbs described in the LU 6.2 architecture.

Context
This macroinstruction can be issued from the SEND or PENDING_SEND conversation state. This
macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
APPCCMD CONTROL=PREPRCV, QUALIFY=DATAFLU

```

APPCCMD

```
CONTROL=PREPRCV, QUALIFY=DATAFLU
```

RPL=rpl_address_field

```
(AADDRESS rpl_address_register)
```

AAREA=rpl_extension_address_field

```
(AADDRESS rpl_extension_address_register)
```

---

**APPCCMD CONTROL=PREPRCV, QUALIFY=DATAFLU**

### Purpose
This macroinstruction sends data to a partner LU and flushes the SEND buffer. The conversation state for
the application program is then changed from SEND to RECEIVE.

### Usage
This macroinstruction combines the functions of two macroinstructions, APPCCMD CONTROL=SEND,
QUALIFY=DATA followed by APPCCMD CONTROL=PREPRCV, QUALIFY=FLUSH. VTAM sends any data
currently in the SEND buffer. This data is followed by the data specified on the macroinstruction to the
partner LU. The application program must ensure that the data sent completes a logical record.

If the data is sent successfully, the conversation is placed in RECEIVE state. The conversation state is
found in the CONSTATE field when the macroinstruction completes.

This macroinstruction corresponds to the SEND_DATA followed by PREPARE_TO_RECEIVE (TYPE=FLUSH)
verbs described in the LU 6.2 architecture.

### Context
This macroinstruction can be issued from the SEND or PENDING_SEND conversation state. This
macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

### Syntax
```line
name

APPCCMD CONTROL=PREPRCV, QUALIFY=DATAFLU

DATAFLU

RPL=rpl_address_field

(AADDRESS rpl_address_register)

AAREA=rpl_extension_address_field

(AADDRESS rpl_extension_address_register)
```
ACB = acb_address_field
    (acb_address_register)

AREA = data_area_or_buffer_list_address_field
    (data_area_or_buffer_list_address_register)

BRANCH = NO
    YES

CONMODE = BUFFCA
    CS
    LLCA
    SAME

CONVID = 32-bit_resource_id_field
    (32-bit_resource_id_register)

CONXMOD = CA
    CS
    SAME

CRYPT = NO
    YES

ECB = INTERNAL
    (ecb_address_field)
    (ecb_address_register)

EXIT = exit_routine_address_field
    (exit_routine_address_register)
,**OPTCD** = (6
ASY 3
SYN 3
BUFFLST 3
NBUFFLST 3
XBUFLST 3
KEEPSRB 3 7
NKEEPSRB 3 8)

,**RECLEN** = (data_length 3
(data_length_register))

**Notes:**

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

**Input parameters**

The following information shows descriptions of the input parameters:

**AAREA=**rpl_extension_address_field

*AAREA=*(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**acb_address_field

*ACB=*(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=**data_area_or_buffer_list_address_field

*AREA=*(data_area_or_buffer_list_address_register)

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this
conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.
CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=events_address
ECB=(events_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

EXIT=exit_routine_address
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=BUFFLST**

Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

**OPTCD=NBUFFLST**

Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

**OPTCD=XBUFLST**

Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of 'X'002C', 'X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

**RECLN=**data_length**

**RECLN=(**data_length_register**)**

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLN specifies the length of the buffer list that in turn points to the data to be sent. RECLN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)
RPL=:\texttt{rpl\_address\_field}\n
\textbf{RPL=(\texttt{rpl\_address\_register})}\n
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

\textbf{RPL and RPL extension fields modified by macroinstruction}\n
The following information shows descriptions of RPL and RPL extension fields:

\textbf{CONSTATE}\n
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. It can have the following values:

- X'01'\n  SEND\n- X'02'\n  RECEIVE\n- X'03'\n  RECEIVE\_CONFIRM\n- X'04'\n  RECEIVE\_CONFIRM\_SEND\n- X'05'\n  RECEIVE\_CONFIRM\_DEALLOCATE\n- X'07'\n  PENDING\_END\_CONVERSATION\_LOG\n- X'08'\n  END\_CONVERSATION\n- X'09'\n  PENDING\_SEND\n- X'0A'\n  PENDING\_RECEIVE\_ONLY\_LOG\n
\textbf{EXPDLEN}\n
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

\textbf{EXPDRCV}\n
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

\textbf{FDB2}\n
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

\textbf{FMH5LEN}\n
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

\textbf{FMH5RCV}\n
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

\textbf{YES (B'1')}\n
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.
NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.
LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:
X'0004'
  ALLOCATION_ERROR
X'0014'
  DEALLOCATE_ABEND_PROGRAM
X'0018'
  DEALLOCATE_ABEND_SERVICE
X'001C'
  DEALLOCATE_ABEND_TIMER
X'0030'
  PROGRAM_ERROR_NO_TRUNC
X'0034'
  PROGRAM_ERRORPURGING
X'0038'
  PROGRAM_ERROR_TRUNC
X'003C'
  SERVICE_ERROR_NO_TRUNC
X'0040'
  SERVICE_ERROR_PURGING
X'0044'
  SERVICE_ERROR_TRUNC
X'005C'
  USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL,
the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and
not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine
if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS
Communications Server: SNA Programmer's LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application
program. It is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL
extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also
can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the
session for the conversation was deactivated. Not all RCPRI values have sense data associated with
them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an
FMH-7 sense code that VTAM did not recognize.

**SIGDATA**

The field in the RPL extension in which the signal code and signal extension fields of a received
SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES.
This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote
application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation
(APPCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified
RTSRTRN=EXPD.

**SIGRCV**

The field in the RPL extension that returns an indication of whether an application program's partner
has requested permission to send. It is labeled RPL6RSIG in the RPL extension. This field and the
SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2
architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation
(APPCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified
RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

**YES (B'1')**

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and
signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA
field.

**NO (B'0')**

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field
contains no meaningful information.

**STSHBF**

The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to
give the current position (address and displacement) in the application-supplied data buffer or buffer
list entry (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs
while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in
the RPL extension.

**STSHDS**

The field in the RPL extension that returns the displacement into the current buffer. It is used with
STSHBF to give the current position (address and displacement) in the application-supplied data
buffer or buffer list entry (the area pointed to by the AREA field of the RPL) when a temporary storage
shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

These changes are applicable when RCPRI indicates OK.

The conversation state is RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>RC PRI</td>
<td>RC SEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>CONTROL/QUALIFY_VALUE_NOT_VALID_FOR_FULL-DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0002'</td>
<td>CSM_DETECTED_ERROR—INVALID_BUFFER_TOKEN_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0003'</td>
<td>CSM_DETECTED_ERROR—INVALID_INSTANCE_ID_SPECIFIED</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=PREPRCV, QUALIFY=FLUSH**

**Purpose**

This macroinstruction flushes the application program's SEND buffer and changes the conversation state from SEND to RECEIVE.
**Usage**

This macroinstruction executes the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction. The application program must ensure that the data in the SEND buffer completes a logical record.

If the data is sent successfully, the conversation is put in RECEIVE state. The conversation state is in the CONSTATE field when the macroinstruction completes.

This macroinstruction corresponds to the PREPARE_TO_RECEIVE (TYPE=FLUSH) verb described in the LU 6.2 architecture.

**Context**

This macroinstruction can be issued from the SEND or PENDING_SEND conversation state. This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

\[
\text{name} \quad \text{APPCCMD} \quad \text{CONTROL=} \quad \text{PREPRCV} \quad \text{QUALIFY=} \quad \text{FLUSH} \\
\text{RPL} = \quad \text{rpl_address_field} \\
\text{AAREA} \quad \text{rpl_extension_address_field} \\
\text{ACB} = \quad \text{acb_address_field} \\
\text{BRANCH} \quad \text{NO} \quad \text{YES} \\
\text{CONMODE} \quad \text{BUFFCA} \quad \text{CS} \quad \text{LLCA} \quad \text{SAME} \\
\text{CONVID} \quad \text{32-bit_resource_id_field} \\
\]
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPsRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
BRANCH
Specifies whether authorized path processing is to be used for application programs running in 
supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use 
BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of 
the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state 
(non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather 
than under a TCB, the macroinstruction is processed in this manner automatically, regardless of 
the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is 
labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
 Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that 
APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the 
application program is to receive data independently of the logical-record format of the data. 
BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC| 
ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only 
APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this 
conversation. When the application program issues APPCCMD CONTROL=RECEIVE, 
QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-
record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that 
APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this 
conversation and that the application program is to receive data in terms of the logical-record 
format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, 
QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
 Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is 
labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
 Specifies that the mode for expedited information is to be put in such a state that expedited 
information can be received by either a specific-type macroinstruction or an any-type 
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD 
CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited 
information can be received only by a specific-type macroinstruction, such as, APPCCMD 
CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
 Specifies that the conversation mode for expedited information is to remain unchanged at the 
completion of this macroinstruction.
ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. It can have the following values:

X’01’
SEND
X’02’
RECEIVE
X’03’
RECEIVE_CONFIRM
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
These changes are applicable when RCPRI indicates OK.
The conversation state is RECEIVE after successful processing. See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED VECTOR PROVIDED ON APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED—CONTROL/QUALIFY_VALUE_NOT_VALID_FOR_FULL-DUPEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY**

### Purpose

This macroinstruction receives expedited information from any active conversation whose expedited information mode is continue-any. VTAM will wait for expedited information to arrive on a conversation in continue-any mode to satisfy the macroinstruction request.

### Usage

This macroinstruction can be used when the application program is maintaining multiple asynchronous conversations. Instead of issuing APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC for each
conversation, the application program can put the conversations in continue-any mode for receiving expedited information and issue a single APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY.

The application program must receive the entire amount of expedited data available. If the length of the area specified by the application is not sufficient to receive all the expedited data available, an RCPRI,RCSEC combination of PARAMETER_ERROR_SUPERED_LENGTH_INSUFFICIENT is returned to the application. The maximum amount of data that can be received from the partner is 86 bytes.

A Request_To_Send_Received indication is sufficient to complete this macroinstruction. If a Request_To_Send_Received indication and expedited data are present, then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on a conversation whose expedited information mode is continue-any, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction. The conversation identifier of the conversation that satisfied the macroinstruction is placed in the CONVID field.

Multiple APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY macroinstructions can be outstanding concurrently. The macroinstruction can be issued when no conversations exist that are in continue-any mode for receiving expedited information. VTAM queues the APPCCMD until one or more conversations are placed in continue-any mode for receiving information and has expedited information available to be received.

**Context**

Input states are not applicable to this macroinstruction. Only expedited information for a conversation that is not in PENDING_DEALLOCATE, END_CONV or FDX_RESET state and whose expedited information mode is continue-any satisfies this type of RCVEXPD.

**Syntax**

```
name  APPCCMD --- CONTROL --- = RCVEXPD --- , --- QUALIFY --- = ANY

, --- RPL = rpl_address_field

, --- AAREA = rpl_extension_address_field

, --- ACB = acb_address_field

, --- AREA = data_area_address_field
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.
Input parameters
The following information shows descriptions of the input parameters:

AAREA=
AAREA=(rpl_extension_address_field)
   Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=
ACB=(acb_address_field)
   Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=
AREA=(data_area_address_field)
   Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

AREALEN=
AREALEN=(data_area_length)
   Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBUFL in the RPL.

BRANCH
   Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

   BRANCH=NO
   Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

   BRANCH=YES
   Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
   Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

   CONMODE=BUFFCA
   Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

   CONMODE=CS
   Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

   CONMODE=LLCA
   Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data.
format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. For half-duplex conversations, this field can have the following values:
- X'01'
  SEND
- X'02'
  RECEIVE
- X'03'
  RECEIVE_CONFIRM
- X'04'
  RECEIVE_CONFIRM_SEND
- X'05'
  RECEIVE_CONFIRM_DEALLOCATE
- X'06'
  PENDING_DEALLOCATE
- X'07'
  PENDING_END_CONVERSATION_LOG
- X'08'
  END_CONVERSATION
- X'09'
  PENDING_SEND
- X'0A'
  PENDING_RECEIVE_LOG
For full-duplex conversations, this field can have the following values:
- X'80'
  FDX_RESET
- X'81'
  SEND/RECEIVE
- X'82'
  SEND_ONLY
- X'83'
  RECEIVE_ONLY
- X'84'
  PENDING_SEND/RECEIVE_LOG
- X'85'
  PENDING_RECEIVE-ONLY_LOG
- X'86'
  PENDING_RESET_LOG
CONVID
Specifies the resource identifier of the conversation on which information was received. A value is placed in this field by VTAM only if QUALIFY=ANY. This field is labeled RPL6CNVD in the RPL extension.

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMHS set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCEVFHM5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLLEN
The field in the RPL that returns to the application program the actual amount of expedited data the application program received. If the application program receives information other than data, this variable is set to 0. This field is labeled RPLRLLEN in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.
X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.
SIGRCV

The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

YES (B'1')

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes

No state changes are associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RCVEXP, QUALIFY=IANY

Purpose

This macroinstruction receives expedited information from any active conversation whose expedited information mode is continue-any. VTAM will not wait for expedited information to arrive on a conversation in continue-any mode to satisfy the macroinstruction.

Usage

This macroinstruction can be used when the application program is maintaining multiple asynchronous conversations. Instead of issuing APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC for each conversation, the application program can put the conversations in continue-any mode for receiving expedited information and issue a single APPCCMD CONTROL=RCVEXP, QUALIFY=IANY.

A Request_To_Send_Received indication is sufficient to successfully complete this macroinstruction. If a Request_To_Send_Received indication and expedited data are present then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on a conversation whose expedited information mode is continue-any, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction. The conversation identifier of the conversation that satisfied the macroinstruction is placed in the CONVID field.

When issued and no conversation exists in a continue-any mode for expedited data or no conversations in continue-any mode have received expedited information, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application program.

The application must receive the entire amount of expedited data available. If the length of the area specified by the application is not sufficient to receive all the expedited data available, an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application.

Context

Input states are not applicable to this macroinstruction. Only expedited information for a conversation that is not in PENDING_DEALLOCATE, END_CONV, or FDX_RESET and whose expedited information mode is continue-any satisfies this type of RCVEXP.

Syntax

```plaintext
APPCCMD CONTROL=RCVEXP QUALIFY=IANY
```
AAREA = rpl_extension_address_field
   ( rpl_extension_address_register )

ACB = acb_address_field
   ( acb_address_register )

AREA = data_area_address_field
   ( data_area_address_register )

AREALEN = data_area_length
   ( data_area_length_register )

BRANCH = NO
   YES

CONMODE = BUFFCA
   CS
   LLCA
   SAME

CONXMOD = CA
   CS
   SAME

ECB = INTERNAL
   ecb_address_field
   ( ecb_address_register )

EXIT = exit_routine_address_field
   ( exit_routine_address_register )

OPTCD = ( ASY
   SYN
   KEEPSRB
   NKEEPSRB )
Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=data_area_address_field**

AREA=(data_area_address_register)

Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

**AREALEN=data_area_length**

AREALEN=(data_area_length_register)

Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.
**CONMODE=BUFFCA**
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the conversation mode of the conversation is to remain unchanged.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

- **CONXMOD=CA**
  Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

- **CONXMOD=CS**
  Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

- **CONXMOD=SAME**
  Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

- **ECB=INTERNAL**
  Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

- **ECB=ecb_address_field**
  Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

- **EXIT=exit_routine_address_field**
  Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.
For half-duplex conversations, this field can have the following values:

X'01'
SEND
X'02'
RECEIVE
X'03'
RECEIVE_CONFIRM
X'04'
RECEIVE_CONFIRM_SEND
X'05'
RECEIVE_CONFIRM_DEALLOCATE
X'06'
PENDING_DEALLOCATE
X'07'
PENDING_END_CONVERSATION_LOG
X'08'
END_CONVERSATION
X'09'
PENDING_SEND
X'0A'
PENDING_RECEIVE_LOG
For full-duplex conversations, this field can have the following values:
CONVID
Specifies the resource identifier of the conversation on which information was received. A value is placed in this field by VTAM only if QUALIFY=ANY|IANY. This field is labeled RPL6CNVD in the RPL extension.

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B’1’)
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B’0’)
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
The field in the RPL that returns to the application program the actual amount of expedited data the application program received up to the maximum. If the application program receives information other than data, this variable is set to 0. This field is labeled RPLRLEN in the RPL.

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

Specifies 4 bytes of user data that the application program requests be associated with a conversation.Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

No state changes are associated with this macroinstruction.

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RCVEXPD, QUALIFY=ISPEC

**Purpose**
This macroinstruction receives expedited information immediately available on a specified conversation. VTAM will not wait for expedited information to arrive to satisfy the macroinstruction request.

**Usage**
A Request_To_Send_Received indication is sufficient to successfully complete this macroinstruction. The conversation mode (CONXMOD) for expedited data may be either CA or CS. If a Request_To_Send_Received indication and expedited data are present then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on the conversation, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction.

If expedited information is not available, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application.

The application must receive the entire amount of expedited data available. If the length of the area specified by the application is not sufficient to receive all the expedited data available, an RCPRI, RCSEC combination of X'0002C', X'00008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application.

If this macroinstruction is issued while another RCVEXPD macroinstruction is currently outstanding for the specified conversation, an RCPRI, RCSEC combination of X'0002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application. The maximum amount of expedited data that can be received is 86 bytes.

If the RECEIVE EXPEDITED queue has been prohibited, then an RCPRI, RCSEC combination of X'00A0', X'00002', REQUEST_NOT_ALLOWED—REQUEST_BLOCKED is returned to the application. The RECEIVE EXPEDITED queue is prohibited when the conversation is in the process of being deallocated or terminated.

If the macroinstruction is issued for a conversation in PENDING_DEALLOCATE state, an RCPRI, RCSEC combination of X'0050', X'00000', STATE_ERROR is returned to the conversation.

If the conversation ends before this macroinstruction can process, an RCPRI, RCSEC combination of X'00000', X'00009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application.
This macroinstruction corresponds to the RECEIVE_EXPEDITED_DATA (IMMEDIATE) verb described in the LU 6.2 architecture.

**Context**

This macroinstruction can be issued from any conversation state except PENDING_DEALLOCATE, END_CONV, or FDX_RESET.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
name          = APPCCMD —— CONTROL —— RCVEXPD —— QUALIFY —— ISPEC
             
, — RPL —— = rpl_address_field
             ( — rpl_address_register — )

, — AAREA —— = rpl_extension_address_field
             ( — rpl_extension_address_register — )

, — ACB —— = acb_address_field
             ( — acb_address_register — )

, — AREA —— = data_area_address_field
             ( — data_area_address_register — )

, — AREALEN —— = data_area_length
             ( — data_area_length_register — )

, — BRANCH —— = NO
             YES

, — CONMODE —— = BUFFCA
             CS
             LLCA
             SAME

, — CONVID —— = 32-bit_resource_id_field
             ( — 32-bit_resource_id_register — )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPESRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=data_area_address_field
AREA=(data_area_address_register)
Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

AREALEN=data_area_length
AREALEN=(data_area_length_register)
Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ech_address_field**
ECB=(ech_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**
The following information shows descriptions of RPL and RPL extension fields:
**CONSTATE**
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- **X'01'**: SEND
- **X'02'**: RECEIVE
- **X'03'**: RECEIVE_CONFIRM
- **X'04'**: RECEIVE_CONFIRM_SEND
- **X'05'**: RECEIVE_CONFIRM_DEALLOCATE
- **X'06'**: PENDING_DEALLOCATE
- **X'07'**: PENDING_END_CONVERSATION_LOG
- **X'08'**: END_CONVERSATION
- **X'09'**: PENDING_SEND
- **X'0A'**: PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- **X'80'**: FDX_RESET
- **X'81'**: SEND/RECEIVE
- **X'82'**: SEND_ONLY
- **X'83'**: RECEIVE_ONLY
- **X'84'**: PENDING_SEND/RECEIVE_LOG
- **X'85'**: PENDING_RECEIVE-ONLY_LOG
- **X'86'**: PENDING_RESET_LOG

**EXPDLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.
**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLLEN**

The field in the RPL that returns to the application program the actual amount of expedited data the application program received. The value returned will always be less than or equal to the value specified for AREALEN. This value will be set to 0 if the macroinstruction is being completed because of a REQUEST_TO_SEND being received.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SIGDATA**

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**SIGRCV**

The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL.

**YES (B'1')**

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.
USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
There are no state changes caused by the execution of this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC

**Purpose**

This macroinstruction receives expedited information from the specified conversation. VTAM will wait for expedited information to arrive to satisfy the macroinstruction request. If expedited information is immediately available, then the application receives it without waiting. The expedited information mode may be continue-any or continue-specific.

**Usage**

A Request_To_Send_Received indication is sufficient to successfully complete this macroinstruction. If a Request_To_Send_Received indication and expedited data are present then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on the conversation, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction.

The application must receive the entire amount of expedited data available. If the length of the area specified by the application program is not sufficient to receive all the expedited data available, an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR_SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. The maximum amount of expedited data that can be received is 86 bytes.

If this macroinstruction is issued while another RCVEXPD macroinstruction is currently outstanding for the specified conversation, an RCPRI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application.

If the RECEIVE EXPEDITED queue has been prohibited, then an RCPRI, RCSEC combination of X'00A0', X'0002', REQUEST_NOT_ALLOWED—REQUEST_BLOCKED is returned to the application. The RECEIVE EXPEDITED queue is prohibited when the conversation is in the process of being allocated or terminated.

If the macroinstruction is issued for a half-duplex conversation and a negative response is received from the partner, then an RCPRI, RCSEC combination of X'00A0', X'0003', REQUEST_NOT_ALLOWED—EXECUTION_OF_REQUEST_TERMINATED will be returned to the application.

If the macroinstruction is issued for a conversation in PENDING_DEALLOCATE state, an RCPRI, RCSEC combination of X'0050', X'0000', STATE_ERROR is returned to the application.

If the conversation is terminated before expedited information is received, an RCPRI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application.

This macroinstruction corresponds to the RECEIVE_EXPEDITED_DATA (WHEN_EXPEDITED_DATA_RECEIVED) verb described in the LU 6.2 architecture.

**Context**

This macroinstruction can be issued from any conversation state except PENDING_DEALLOCATE, END_CONV, or FDX_RESET.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```plaintext
APPCCMD name -- CONTROL = RCVEXPD , QUALIFY = SPEC
```
LU 6.2 macroinstruction syntax and operands 277
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2 See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters
The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=data_area_address_field**

AREA=(data_area_address_register)

Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

**AREALEN=data_area_length**

AREALEN=(data_area_length_register)

Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
**BRANCH=NO**
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
 Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
 Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
 Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.
**ECB=ecb_address_field**
**ECB=(ecb_address_register)**
Specifications that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
**EXIT=(exit_routine_address_register)**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**
**RPL=(rpl_address_register)**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- **X’01’**: SEND
- **X’02’**: RECEIVE
- **X’03’**: RECEIVE_CONFIRM
- **X’04’**: RECEIVE_CONFIRM_SEND
- **X’05’**: RECEIVE_CONFIRM_DEALLOCATE
For full duplex conversations, this field can have the following values:

- **X'80'**  
  FDX_RESET

- **X'81'**  
  SEND/RECEIVE

- **X'82'**  
  SEND_ONLY

- **X'83'**  
  RECEIVE_ONLY

- **X'84'**  
  PENDING_SEND/RECEIVE_LOG

- **X'85'**  
  PENDING_RECEIVE-ONLY_LOG

- **X'86'**  
  PENDING_RESET_LOG

**EXPDLLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**  
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**  
  No FMH-5s are waiting to be received by the application program.
**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLLEN**
The field in the RPL that returns to the application program the actual amount of expedited data the application program received. The value returned will always be less than or equal to the value specified for AREALEN. This value is set to 0 if the macroinstruction completes only due to receipt of a REQUEST_TO_SEND indication.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
There are no state changes caused by the execution of this macroinstruction.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.
APPCCMD CONTROL=RCVFMH5, QUALIFY=DATAQUE

Purpose

This macroinstruction receives an FMH-5, which begins the application program's participation in a conversation.

This macroinstruction allows the application to specify how expedited information is received.

Usage

When this macroinstruction is issued, VTAM copies the FMH-5, which represents a new conversation, into the area specified on the AREA parameter. When the macroinstruction completes, the new conversation identifier can be found in the CONVID field. The new conversation will be in RECEIVE state for half-duplex conversations and in SEND/RECEIVE state for full-duplex conversations.

If this macroinstruction is issued before an FMH-5 is received, VTAM waits until the FMH-5 is received to complete the macroinstruction. When an FMH-5 is received, VTAM bypasses the ATTN exit. If VTAM receives the FMH-5 before this macroinstruction is issued, VTAM schedules the ATTN exit. In either case, VTAM then moves the FMH-5 to the application’s buffer and returns the CONVID and other return parameters.
After performing the operation of the RCVFMH5, VTAM examines the setting of the FILL parameter. If FILL=LL has been specified, this macroinstruction performs the functions of an APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC with a FILL=LL. That is, VTAM receives a single logical record. This would be the first logical record after the FMH-5. However, if FILL=BUFF has been specified, this macroinstruction performs the functions of an APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC with a FILL=BUFF. If there is insufficient information to complete the receive, the macroinstruction is suspended until more information is received from the partner. As is normally done for a receive macroinstruction, if any of the following conditions occurs, the APPCCMD CONTROL=RCVFMH5, QUALIFY=DATAQUE completes:

- The local receive buffer is completely filled.
- A complete logical record is received AND FILL=LL was specified.
- A SEND indication is received.
- A CONFIRM indication is received.
- A DEALLOCATE indication is received.
- An ERROR condition is detected.

The application program can use the FMH-5 to perform conversation level security processing. Also, the FMH-5 indicates whether any GDS fields, such as DCE security or program initialization (PIP) data, follows the FMH-5.

Context

This macroinstruction can be issued from the RESET conversation state.

This macroinstruction is not mode-specific and might be issued for a mode that is retained for persistent LU-LU sessions. However, an FMH-5 is not returned for a mode that is being retained for persistent LU-LU sessions when this macroinstruction is issued.

Syntax

```plaintext
APPCCMD name — — CONTROL = — — RCVFMH5 — — QUALIFY = — — DATAQUE — — RPL = rpl_address_field ( rpl_address_register ) — — AAREA = rpl_extension_address_field ( rpl_extension_address_register ) — — ACB = acb_address_field ( acb_address_register ) — — AREA = data_area_address_field ( data_area_address_register )
```
Notes:

1 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=data_area_address_field**

AREA=(data_area_address_register)

Specifies the data area in which the application program is to receive the FMH-5 and any associated data. This field is labeled RPLAREA in the RPL.

**AREALEN=data_area_length**

AREALEN=(data_area_length_register)

Specifies the size of the supplied buffer area. An FMH-5 is, at most, 255 bytes in length. Because the application cannot determine the length of the FMH-5 when the RCVFMH5 request is queued, VTAM fails this macroinstruction if the length of AREALEN is less than 255 with an RCPRI, RCSEC combination of X'002C', X'0008'. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use
BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH= YES**
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CD**
Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data. This parameter is valid only for half-duplex conversations.

**CD=DEFER**
Specifies that the conversation state will be in PEND_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CD=IMMED**
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CONMODE**
Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.
ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=user_data_register
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the
conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
    Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:
    • VTRINA=0
    • The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
    • The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
    Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CGID
    Specifies the 32-bit conversation group identifier.

CONVID
    The field in the RPL extension that returns the resource identifier of the new conversation. This field is labeled RPL6CNVD in the RPL extension.

CONSTATE
    The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST.

For half-duplex conversations, this field can have the following values:

X'00'
    RESET
X'01'
    SEND
X'02'
    RECEIVE
X'03'
    RECEIVE_CONFIRM
X'04'
    RECEIVE_CONFIRM_SEND
X'05'
    RECEIVE_CONFIRM_DEALLOC
X'07'
    PENDING_END_CONVERSATION_LOG
X'08'
    END_CONVERSATION
X'09'
    PENDING_SEND
For full-duplex conversations, this field can have the following values.

- **X'80'**
  - FDX_RESET
- **X'81'**
  - SEND/RECEIVE
- **X'82'**
  - SEND_ONLY
- **X'83'**
  - RECEIVE_ONLY
- **X'84'**
  - PENDING_SEND/RECEIVE_LOG
- **X'85'**
  - PENDING_RECEIVE_ONLY_LOG
- **X'86'**
  - PENDING_RESET_LOG

**CRYPTLVL**
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

- **NONE (B'00')**
  - No data is to be encrypted.
- **SELECTIVE (B'01')**
  - The application program specifies the data that is to be encrypted.
- **REQUIRED (B'11')**
  - All data is to be encrypted.

**EXPDLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether another FMH-5, other than the one currently being passed to the application program on this APPCCMD, has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  - One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.
- **NO (B'0')**
  - No other FMH-5s are waiting to be received by the application program.
LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B’1’)
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:
- `X'0004'`
  ALLOCATION_ERROR
- `X'0014'`
  DEALLOCATE_ABEND_PROGRAM
- `X'0018'`
  DEALLOCATE_ABEND_SERVICE
- `X'001C'`
  DEALLOCATE_ABEND_TIMER
- `X'0030'`
  PROGRAM_ERROR_NO_TRUNC
- `X'0034'`
  PROGRAM_ERROR_PURGING
- `X'0038'`
  PROGRAM_ERROR_TRUNC
- `X'003C'`
  SERVICE_ERROR_NO_TRUNC
- `X'0040'`
  SERVICE_ERROR_PURGING
- `X'0044'`
  SERVICE_ERROR_TRUNC
- `X'005C'`
  USER_ERROR_CODE_RECEIVED

NO (B’0’)
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

LOGMODE
The field in the RPL extension that returns the logon mode name of the session over which the FMH-5 is being returned on this APPCCMD macroinstruction. It is an 8-byte name, padded on the right with blanks. It is labeled RPL6MODE.

LUNAME
The field in the RPL extension that returns the name of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID
The field in the RPL extension that returns the network identifier of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction.

This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.
RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLLEN
The field in the RPL that returns to the application the size, in bytes, of the FMH-5. This field is labeled RPLRLEN in the RPL. If the RCPRI field equals X'0000', (OK), RECLLEN specifies the number of bytes of the supplied AREA field that were used to return the FMH-5 to the application program.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 4-byte sense code. This sense code has meaning if the RCPRI return code indicates a resource failure problem. It is labeled RPL6SNSI. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated.

SESSID
The field in the RPL extension that, when SESSIDL is not equal to 0, returns a session instance identifier for the session over which the FMH-5 was received. The format of the session instance identifier is described in the z/OS Communications Server: SNA Programmer's LU 6.2 Guide This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. It is labeled RPL6RSIG. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.
NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

SLS
The field in the RPL extension that indicates whether or not the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

WHATRCV
The field in the RPL extension that returns a mask specifying what the application program received. It is labeled RPL6WHAT. The application program should examine this WHATRCV mask only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format.

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>Meaning</th>
<th>RPLRCV2</th>
<th>Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DATA</td>
<td>0</td>
<td>PARTIAL_PS_HEADER</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
<td>1–7</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X'8000'. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a discussion of the meaning of this field.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- Partner's DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner's nonce vector (X'14')
- PCID vector (X'17')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')
State changes
These changes are applicable when RCPRI indicates OK.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS PGM NOT AVAIL NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS PGM NOT AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT TRANS PGM NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT NOT SUPPORTED BY PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=RCVFMH5, QUALIFY=NULL

**Purpose**
This macroinstruction receives an FMH-5, which begins the application program's participation in a conversation. This macroinstruction allows the application to specify how expedited information is received.

**Usage**
When this macroinstruction is issued, VTAM copies the FMH-5, which represents a new conversation, into the area specified on the AREA parameter. When the macroinstruction completes, the new conversation identifier can be found in the CONVID field. The new conversation will be in RECEIVE state for half-duplex conversations and in SEND/RECEIVE state for full-duplex conversations.

The application program can use the FMH-5 to perform conversation level security processing. Also, the FMH-5 indicates whether any GDS fields, such as DCE security or program initialization (PIP) data, follows the FMH-5. If so, the application program should issue APPCCMD CONTROL=RECEIVE to receive the GDS data.

If no FMH-5 is available for the application to receive, this macroinstruction is rejected with an RCPRI return code of X'0060'.

For information on how the application program is informed that an FMH-5 is ready to be received, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**Context**
This macroinstruction can be issued from the RESET conversation state.

This macroinstruction is not mode-specific and might be issued for a mode that is retained for persistent LU-LU sessions. However, an FMH-5 is not returned for a mode that is being retained for persistent LU-LU sessions when this macroinstruction is issued.

**Syntax**

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_ORRESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
APPCCMD — CONTROL = RCVFMH5

, QUALIFY = NULL

, RPL = rpl_address_field

, AAREA = rpl_extension_address_field

, ACB = acb_address_field

, AREA = data_area_address_field

, AREALEN = data_area_length

, BRANCH = NO

, CONMODE = BUFFCA

, CONXMOD = CA

, ECB = ecb_address_field

, EXIT = exit_routine_address_field
Notes:

1 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

2 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

3 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

4 ECB is meaningful only for asynchronous operations.

5 EXIT is meaningful only for asynchronous operations.

6 You can code more than one suboperand on OPTCD, but no more than one from each group.

7 KEEPSRB is meaningful only for synchronous operations.

8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with
transmission programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=** data_area_address_field

**AREA=**(data_area_address_register)

Specifies the data area in which the application program is to receive the FMH-5. This field is labeled RPLAREA in the RPL.

**AREALEN=** data_area_length

**AREALEN=**(data_area_length_register)

Specifies the size of the supplied buffer area. The supplied buffer area must be large enough to contain the entire FMH-5. An FMH-5 is at most 255 bytes in length (it has only 1 byte for a length count). If a 255-byte buffer is used to receive the FMH-5, the RCVFMH5 macroinstruction will never fail for lack of buffer space. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=**BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=**CS

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=**LLCA

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=**CA

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

LU 6.2 macroinstruction syntax and operands 299
USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
   Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
   Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:
   • VTRINA=0
   • The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
   • The value for VTRINL is not specified.
   This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
   Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CGID
   Specifies the 32-bit conversation group identifier. It is labeled RPL6CGID in the RPL extension.

CONSTATE
   The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.
   For half-duplex conversations, this field can have the following values:
   X'00'
      RESET
   X'02'
      RECEIVE
   X'08'
      END_CONVERSATION
   For full-duplex conversations, this field can have the following values.
   X'00'
      RESET
   X'80'
      FDX_RESET
   X'81'
      SEND/RECEIVE

CONVID
   The field in the RPL extension that returns the resource identifier of the new conversation. This field is labeled RPL6CNVD in the RPL extension.
CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether another FMH-5, other than the one currently being passed to the application program on this APPCCMD, has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No other FMH-5s are waiting to be received by the application program.

LOGMODE
The field in the RPL extension that returns the logon mode name of the session over which the FMH-5 is being returned on this APPCCMD macroinstruction. It is an 8-byte name, padded on the right with blanks. It is labeled RPL6MODE in the RPL extension.

LUNAME
The field in the RPL extension that returns the name of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID
The field in the RPL extension that returns the network identifier of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction.

This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
RECLEN
The field in the RPL that returns to the application the size, in bytes, of the FMH-5. This field is labeled RPLRLEN in the RPL. If the RCPRI field equals X'0000', (OK), RECLEN specifies the number of bytes of the supplied AREA field that were used to return the FMH-5 to the application program. If the (RCPRI, RCSEC) fields equal X'0002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT, it indicates the size of the FMH-5. However, in the latter case, because the supplied buffer was not large enough to contain the entire FMH-5, the FMH-5 is not returned to the application program. The application program is informed, through the FMH5LEN, of how large the buffer must be in order to receive the FMH-5.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 4-byte sense code. This sense code has meaning if the RCPRI return code indicates a resource failure problem. It is labeled RPL6SNSI in the RPL extension. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated.

SESSID
The field in the RPL extension that, when SESSIDL is not equal to 0, returns a session instance identifier for the session over which the FMH-5 was received. The format of the session instance identifier is described in z/OS Communications Server: SNA Programmer's LU 6.2 Guide. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether or not the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
• VTAM-to-APPL required information vector (X'10')
• Partner's DCE capabilities vector (X'12')
• Local nonce vector (X'13')
• Partner's nonce vector (X'14')
• PCID vector (X'17')
• Session information vector (X'19')
• Partner's application capabilities vector (X'1A')

State changes
These changes are applicable when RCPRI indicates OK.
For half-duplex conversations, the conversation state is RECEIVE after successful processing.
For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.
See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.
### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREALENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0060'</td>
<td>X'0000'</td>
<td>NO_FMH5_AVAILABLE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_ORRESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A4'</td>
<td>X'0000'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

### APPCCMD CONTROL=RCVFMH5, QUALIFY=QUEUE

**Purpose**

This macroinstruction receives an FMH-5, which begins the application program's participation in a conversation.

This macroinstruction allows the application to specify how expedited information is received.

**Usage**

When this macroinstruction is issued, VTAM copies the FMH-5, which represents a new conversation, into the area specified on the AREA parameter. When the macroinstruction completes, the new conversation identifier can be found in the CONVID field. The new conversation will be in RECEIVE state for half-duplex conversations and in SEND/RECEIVE state for full-duplex conversations.

If this macroinstruction is issued before an FMH-5 is received, VTAM waits for the FMH-5 to complete the macroinstruction. When an FMH-5 is received, VTAM bypasses the ATTN exit. If VTAM receives the FMH-5 before this macroinstruction is issued, VTAM schedules the ATTN exit. In either case, VTAM then moves...
the FMH-5 to the application's buffer and returns the CONVID and other return parameters. VTAM retains any data that accompanies the FMH-5.

The application program can use the FMH-5 to perform conversation level security processing. Also, the FMH-5 indicates whether any GDS fields, such as DCE security or program initialization (PIP) data, follows the FMH-5. If so, the application program should issue APPCCMD CONTROL=RECEIVE to receive the PIP data.

For information on how the application program is informed that an FMH-5 is ready to be received, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**Context**

This macroinstruction can be issued from the RESET conversation state.

This macroinstruction is not mode-specific and might be issued for a mode that is retained for persistent LU-LU sessions. However, an FMH-5 is not returned for a mode that is being retained for persistent LU-LU sessions when this macroinstruction is issued.

**Syntax**

```plaintext
APPCCMD  ---  CONTROL  ---  RCVFMH5  ---  QUALIFY  ---  =

- QUEUE

- RPL  =  rpl_address_field

- AAREA  =  rpl_extension_address_field

- ACB  =  acb_address_field

- AREA  =  data_area_address_field

- AREALEN  =  data_area_length

- BRANCH  =  NO
```

304  z/OS Communications Server: SNA Programmer's LU 6.2 Reference
Notes:

1 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

4 ECB is meaningful only for asynchronous operations.

5 EXIT is meaningful only for asynchronous operations.

6 You can code more than one suboperand on OPTCD, but no more than one from each group.

7 KEEPSRB is meaningful only for synchronous operations.

8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** `rpl_extension_address_field`

AAREA=(`rpl_extension_address_register`) Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** `acb_address_field`

ACB=(`acb_address_register`) Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=** `data_area_address_field`

AREA=(`data_area_address_register`) Specifies the data area in which the application program is to receive the FMH-5. This field is labeled RPLAREA in the RPL.

**AREALEN=** `data_area_length`

AREALEN=(`data_area_length_register`) Specifies the size of the supplied buffer area. An FMH-5 is, at most, 255 bytes in length. Because the application cannot determine the length of the FMH-5 when the RCVFMH5 request is queued, VTAM fails this macroinstruction if the length of AREALEN is less than 255 with an RCPRI, RCSEC combination of X'002C', X'0008'. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.
**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**
**ECB=(ecb_address_register)**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
**EXIT=(exit_routine_address_register)**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CGID
Specifies the 32-bit conversation group identifier. It is labeled RPL6CGID in the RPL extension.
CONVID
The field in the RPL extension that returns the resource identifier of the new conversation. This field is labeled RPL6CNVD in the RPL extension.

CONSTATE
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST.
For half-duplex conversations, this field can have the following values:
- X'00' (RESET)
- X'02' (RECEIVE)
- X'08' (END_CONVERSATION)
For full-duplex conversations, this field can have the following values:
- X'00' (RESET)
- X'80' (FDX_RESET)
- X'81' (SEND/RECEIVE)

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.
- NONE (B'00')
  No data is to be encrypted.
- SELECTIVE (B'01')
  The application program specifies the data that is to be encrypted.
- REQUIRED (B'11')
  All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether another FMH-5, other than the one currently being passed to the application program on this APPCCMD, has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.
- YES (B'1')
  One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.
- NO (B'0')
  No other FMH-5s are waiting to be received by the application program.
LOGMODE
The field in the RPL extension that returns the logon mode name of the session over which the FMH-5 is being returned on this APPCCMD macroinstruction. It is an 8-byte name, padded on the right with blanks. It is labeled RPL6MODE.

LUNAME
The field in the RPL extension that returns the name of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID
The field in the RPL extension that returns the network identifier of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction.

This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLLEN
The field in the RPL that returns to the application the size, in bytes, of the FMH-5. This field is labeled RPLRLEN in the RPL. If the RCPRI field equals X'0000', (OK), RECLLEN specifies the number of bytes of the supplied AREA field that were used to return the FMH-5 to the application program.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 4-byte sense code. This sense code has meaning if the RCPRI return code indicates a resource failure problem. It is labeled RPL6SNSI. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated.

SESSID
The field in the RPL extension that, when SESSIDL is not equal to 0, returns a session instance identifier for the session over which the FMH-5 was received. The format of the session instance identifier is described in the z/OS Communications Server: SNA Programmer's LU 6.2 Guide. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.
Vectors returned

VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X'10')
- Partner's DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner's nonce vector (X'14')
- PCID vector (X'17')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is RECEIVE after successful processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>VECTOR AREA NOT VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>VECTOR AREA LENGTH INSUFFICIENT</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A4'</td>
<td>X'0000'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RECEIVE, QUALIFY=ANY

**Purpose**
This macroinstruction receives normal information on a conversation that is in continue-any mode. Unlike other macroinstructions that are used to receive data, the application program does not specify a partner conversation. Instead, the macroinstruction is associated with the first conversation that is in continue-any mode and that receives data.

**Usage**
This macroinstruction can be used when the application program is maintaining multiple asynchronous conversations. Instead of issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC for each conversation, the application program can put the conversations in continue-any mode and issue a single APPCCMD CONTROL=RECEIVE, QUALIFY=ANY.

When VTAM receives data on a continue-any mode conversation, VTAM copies the data into the data area that is specified on the AREA parameter and if sufficient data has been received, then VTAM completes the macroinstruction. The conversation identifier of the conversation that is used to complete the macroinstruction is placed in the CONVID field.

This macroinstruction can be used to receive application program data, conversation status information, and confirmation requests. However, it cannot be used to receive error log information. The application program must use APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to receive error log information.

If VTAM receives notification that a conversation fails on a continue-any mode, this macroinstruction completes with a nonzero return code.

Multiple APPCCMD CONTROL=RECEIVE, QUALIFY=ANY macroinstructions can be outstanding concurrently. The order in which these macroinstructions complete is not necessarily the order in which they were issued. This means that if a conversation is left in continue-any mode, data from multiple RECEIVES could arrive out of order. If the application program cannot detect this and process the data properly, the application program should specify CONMODE=CS on the APPCCMD CONTROL=RECEIVE, QUALIFY=ANY macroinstruction. For more information on specifying CONMODE, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

APPCCMD CONTROL=RECEIVE, QUALIFY=ANY can be issued when no conversations exist that are in continue-any mode and in RECEIVE, SEND/RECEIVE, or RECEIVE-ONLY state. The APPCCMD waits until one or more conversations are placed into continue-any mode and the right state.

An any-mode RECEIVE can lock out a specific-mode RECEIVE. For example, if an application program has issued an any-mode RECEIVE that receives data in terms of buffers, and enough data has not arrived to satisfy the buffer length, VTAM waits until enough data arrives to satisfy the buffer length before honoring a specific mode RECEIVE for the conversation.

This macroinstruction does not directly correspond to any architected verb described in the LU 6.2 architecture.

**Context**
Input states are not applicable to this macroinstruction. Only information for a conversation in RECEIVE, SEND/RECEIVE, or RECEIVE_ONLY state and continue-any mode satisfies this type of RECEIVE.

**Syntax**

---

312  z/OS Communications Server: SNA Programmer's LU 6.2 Reference
LU 6.2 macroinstruction syntax and operands
Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** *rpl_extension_address_field*

**AAREA=**( *rpl_extension_address_register*)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.
ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

AREA=data_area_address_field
AREA=(data_area_address_register)
Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

AREALEN=data_area_length
AREALEN=(data_area_length_register)
Specifies the length value that is the maximum amount of data the application program is to receive.

If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CD
Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PENDING_SEND when a change of direction is received with no data.

CD=DEFER
Specifies that the conversation state will be in PENDING_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CD=IMMED
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.
**CONMODE=BUFFCA**
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**
**ECB=(ecb_address_register)**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
**EXIT=exit_routine_address_register**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NBUFFLST
Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

OPTCD=XBUFFLST
Specifies that the HPDT interface is to be used. VTAM builds an extended buffer list in the address specified by the AREA parameter. Each entry in the buffer list points to a CSM buffer containing the data being received by the application. The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

Note: Application programs running in TCB-mode supervisor state must specify BRANCH=YES for HPDT requests.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

VTROUTA=vector_address_field
VTROUTA=(vector_address_register)
Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFFLST is specified, this field must point to the XBUFFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.)

This field is labeled RPL6VAOA in the RPL extension.

VTROUTL=vector_length_field
VTROUTL=(vector_length_register)
Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. For half-duplex conversations, this field can have the following values:

X'01'
    SEND
For full-duplex conversations, this field can have the following values:

- X'80'  
  FDX_RESET
- X'81'  
  SEND/RECEIVE
- X'82'  
  SEND_ONLY
- X'83'  
  RECEIVE_ONLY
- X'84'  
  PENDING_SEND/RECEIVE_LOG
- X'85'  
  PENDING_RECEIVE-ONLY_LOG
- X'86'  
  PENDING_RESET_LOG

**CONVID**
Specifies the resource identifier of the conversation on which information was received. This field is labeled RPL6CNVD in the RPL extension.

**EXPDLLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.
**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**LOGRCV**
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**YES (B'1')**
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004' ALLOCATION_ERROR
- X'0014' DEALLOCATE_ABEND_PROGRAM
- X'0018' DEALLOCATE_ABEND_SERVICE
- X'001C' DEALLOCATE_ABEND_TIMER
- X'0030' PROGRAM_ERROR_NO_TRUNC
- X'0034' PROGRAM_ERROR_PURGING
- X'0038' PROGRAM_ERROR_TRUNC
- X'003C' SERVICE_ERROR_NO_TRUNC
- X'0040' SERVICE_ERROR_PURGING
- X'0044' SERVICE_ERROR_TRUNC
- X'005C' USER_ERROR_CODE_RECEIVED

**NO (B'0')**
Either no error indicator was received or an error indicator was received but indicated that no log data follows.
**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLLEN**
The field in the RPL that returns to the application program the actual amount of data the application program received up to the maximum. If the application program receives information other than data, this variable is set to 0. When OPTCD=XBUFLST is specified, VTAM returns the actual length of the extended buffer list that is built in the buffer list area pointed to by the AREA operand. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.
USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMLE macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

WHATRCV
The field in the RPL extension that returns a mask specifying what the application program received. It is labeled RPL6WHAT in the RPL extension. The application program should examine this WHATRCV mask only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format:

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>RPL6RCV2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Meaning</td>
</tr>
<tr>
<td>0</td>
<td>DATA</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X'8000'.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a discussion of the meaning of this field. However, LOG_DATA cannot be set on this macroinstruction.

State changes
See the description of the WHATRCV mask for a description of the state changes that occur when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0030'</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_OR_SPECIFIED_INCORRECTLY</td>
</tr>
<tr>
<td>X'003D'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'003A'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT-Requested_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0003'</td>
<td>CSNDETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=RECEIVE, QUALIFY=IANY**

**Purpose**

This macroinstruction receives normal information that is immediately available on a conversation that is in continue-any mode. VTAM does not wait for data to be received before completing this macroinstruction.

**Usage**

When this macroinstruction is issued, VTAM copies all data that is immediately available into the supplied data area or control block that is specified by the AREA parameter. VTAM also returns the identification of the conversation that satisfied the macroinstruction in the CONVID parameter.

This macroinstruction can be used to receive application program data, conversation status information, and confirmation requests. However, it cannot be used to receive error log information. The application program must use APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to receive error log information.

This macroinstruction does not directly correspond to any architected verb described in the LU 6.2 architecture.

If no data is immediately available, an RCPRI,RCSEC of (X'0000', X'0008') NO_INFORMATION_IMMEDIATELY_AVAILABLE is returned to the application.

**Context**

Input states are not applicable to this macroinstruction. Only data for a conversation in RECEIVE, SEND/RECEIVE, or RECEIVE_ONLY state and continue-any mode satisfies this type of RECEIVE.

**Syntax**

```
APPCCMD (name) \nAPPCCMD - CONTROL = RECEIVED, - QUALIFY = - \n              IANY \n              , - RPL = rpl_address_field \n              ( rpl_address_register )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** *rpl_extension_address_field*

**AAREA=(** *rpl_extension_address_register*)*

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** *acb_address_field*

**ACB=(** *acb_address_register*)*

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=data_area_address_field
AREA=(data_area_address_register)
Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

AREALEN=data_area_length
AREALEN=(data_area_length_register)
Specifies the length value that is the maximum amount of data the application program is to receive.

If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=NO to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CD
Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PENDING_SEND when a change of direction is received with no data.

CD=DEFER
Specifies that the conversation state will be in PENDING_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CD=IMMED
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this
conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
- Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**
- Specifies that the continuation mode of the conversation is to remain unchanged.

**CONXMOD**
- Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
- Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

**CONXMOD=CS**
- Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
- Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
- Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
- Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**
- Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
- Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
- Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
- Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
- Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=KEEPSRB
  Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
  Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NBUFFLST
  Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

OPTCD=XBUFLST
  Specifies that the HPDT interface is to be used. VTAM builds an extended buffer list in the address specified by the AREA parameter. Each entry in the buffer list points to a CSM buffer containing the data being received by the application. The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

  Note: Application programs running in TCB-mode supervisor state must specify BRANCH=YES for HPDT requests.

RPL=rpl_address_field
RPL=(rpl_address_register)
  Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

VTROUTA=vector_address_field
VTROUTA=(vector_address_register)
  Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.)

  This field is labeled RPL6VAOA in the RPL extension.

VTROUTL=vector_length_field
VTROUTL=(vector_length_register)
  Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
  The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

  For half-duplex conversations, this field can have the following values:

  X'01'
      SEND
  X'02'
      RECEIVE
  X'03'
      RECEIVE_CONFIRM
  X'04'
      RECEIVE_CONFIRM_SEND
  X'05'
      RECEIVE_CONFIRM_DEALLOCATE
For full-duplex conversations, this field can have the following values:

- X'80' FDX_RESET
- X'81' SEND/RECEIVE
- X'82' SEND_ONLY
- X'83' RECEIVE_ONLY
- X'84' PENDING_SEND/RECEIVE_LOG
- X'85' PENDING_RECEIVE-ONLY_LOG
- X'86' PENDING_RESET_LOG

**CONVID**
Specifies the resource identifier of the conversation on which information was received. A value is placed in this field by VTAM only if QUALIFY=\*ANY. This field is labeled RPL6CNVD in the RPL extension.

**EXPDLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.
NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.
LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004'
  ALLOCATION_ERROR
- X'0014'
  DEALLOCATE_ABEND_PROGRAM
- X'0018'
  DEALLOCATE_ABEND_SERVICE
- X'001C'
  DEALLOCATE_ABEND_TIMER
- X'0030'
  PROGRAM_ERROR_NO_TRUNC
- X'0034'
  PROGRAM_ERROR_PURGING
- X'0038'
  PROGRAM_ERROR_TRUNC
- X'003C'
  SERVICE_ERROR_NO_TRUNC
- X'0040'
  SERVICE_ERROR_PURGING
- X'0044'
  SERVICE_ERROR_TRUNC
- X'005C'
  USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
The field in the RPL that returns to the application program the actual amount of data the application program received up to the maximum. If the application program receives information other than
data, this variable is set to 0. When OPTCD=XBUFLST is specified, VTAM returns the actual length of the extended buffer list that is built in the buffer list area pointed to by the AREA operand. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

**YES (B’1’)**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B’0’)**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**WHATRCV**
The field in the RPL extension that indicates what the application program received. It is labeled RPL6WHAT in the RPL extension. The application program should examine the WHATRCV field only when RCPRI indicates X’0000’. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B’1’) to indicate the type of information that has been received. For instance, if the application program is
being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format:

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>RPL6RCV2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Meaning</td>
</tr>
<tr>
<td>0</td>
<td>DATA</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
</tr>
<tr>
<td>6</td>
<td>PS_HEADER</td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X'8000'.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a discussion of the meaning of this field. However, LOG_DATA cannot be set on this macroinstruction.

State changes
See the description of the WHATRCV mask for a description of the state changes that occur when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0030'</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_OR_SPECIFIED_INCORRECTLY</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERRORPURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT OR DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE.</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RECEIVE, QUALIFY=ISPEC

Purpose
This macroinstruction receives normal information that is immediately available from a specified conversation. The conversation may be in continue-any or continue-specific mode. VTAM does not wait for more data to be received before completing this macroinstruction.

Usage
When this macroinstruction is issued, VTAM copies all data that is immediately available into the supplied data area or control block that is specified by the AREA parameter. The AREALEN parameter specifies the length of the data area. VTAM does not wait to receive any more data before completing the macroinstruction request. If there is no information available, VTAM issues an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION.

When this macroinstruction completes, the RECLEN field indicates how much data was written to the data area. The WHATRCV field indicates what type of data was received.

If VTAM is processing APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY for a conversation and the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=ISPEC for the same conversation, the QUALIFY=ISPEC request fails with an RCPRI, RCSEC combination of X'0000', X'0003', RECEIVE_SPECIFIC_REJECTED. VTAM cannot allow a specific-mode RECEIVE while an any-mode RECEIVE is being processed.

This macroinstruction corresponds to the RECEIVE_IMMEDIATE verb described in the LU 6.2 architecture.

Context
For half-duplex conversations, this macroinstruction can be issued from the following conversation states:
- RECEIVE
- PEND_END_CONV_LOG
- PEND_RCV_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND/RECEIVE
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```plaintext
APPCCMD — CONTROL = RECEIVE , QUALIFY = ISPEC

, RPL = rpl_address_field

(rpl_address_register)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.
**AREA**=data_area_address_field  
**AREA**=(data_area_address_register)

Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

**AREALEN**=data_area_length  
**AREALEN**=(data_area_length_register)

Specifies the length value that is the maximum amount of data the application program is to receive.

If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH**=NO  
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH**=YES  
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CD**

Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.

**CD**=DEFER  
Specifies that the conversation state will be in PEND_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CD**=IMMED  
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE**=BUFFCA  
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC macroinstruction.

**CONMODE**=CS  
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC can be used to receive data on this
conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

FILL
Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter corresponds to FILL=LL|BUFFER described in the LU 6.2 architecture. This field is labeled RPL6FILL in the RPL extension.

FILL=BUFF
Specifies the application program is to receive data independently of its logical-record format, up to the length specified by the AREALEN field of the RPL. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.
**FILL=LL**
Specifies the application program is to receive one logical record, or whatever portion of the logical record is available, up to the length specified by the AREALEN field of the RPL. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NBUFFLST**
Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

**OPTCD=XBUFLST**
Specifies that the HPDT interface is to be used. VTAM builds an extended buffer list in the address specified by the AREA parameter. Each entry in the buffer list points to a CSM buffer containing the data being received by the application. The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

*Note: Application programs running in TCB-mode supervisor state must specify BRANCH=YES for HPDT requests.*

**RPL=**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**VTROUTA=**
Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.)

This field is labeled RPL6VAOA in the RPL extension.

**VTROUTL=**
Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**
The following information shows descriptions of RPL and RPL extension fields:
**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- **X'01'**
  - SEND
- **X'02'**
  - RECEIVE
- **X'03'**
  - RECEIVE_CONFIRM
- **X'04'**
  - RECEIVE_CONFIRM_SEND
- **X'05'**
  - RECEIVE_CONFIRM_DEALLOCATE
- **X'07'**
  - PENDING_END_CONVERSATION_LOG
- **X'08'**
  - END_CONVERSATION
- **X'09'**
  - PENDING_SEND
- **X'0A'**
  - PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- **X'80'**
  - FDX_RESET
- **X'81'**
  - SEND/RECEIVE
- **X'82'**
  - SEND_ONLY
- **X'83'**
  - RECEIVE_ONLY
- **X'84'**
  - PENDING_SEND/RECEIVE_LOG
- **X'85'**
  - PENDING_RECEIVE-ONLY_LOG
- **X'86'**
  - PENDING_RESET_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of
the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**LOGRCV**
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**YES (B'1')**
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004'
  ALLOCATION_ERROR
- X'0014'
  DEALLOCATE_ABEND_PROGRAM
- X'0018'
  DEALLOCATE_ABEND_SERVICE
- X'001C'
  DEALLOCATE_ABEND_TIMER
- X'0030'
  PROGRAM_ERROR_NO_TRUNC
- X'0034'
  PROGRAM_ERROR_PURGING
- X'0038'
  PROGRAM_ERROR_TRUNC
- X'003C'
  SERVICE_ERROR_NO_TRUNC
- X'0040'
  SERVICE_ERROR_PURGING
- X'0044'
  SERVICE_ERROR_TRUNC
- X'0048'
  RESOURCE_FAILURE,_NO_RETRY
- X'0050'
  USER_ERROR_CODE_RECEIVED
NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLLEN**
The field in the RPL that returns to the application program the actual amount of data the application program received. If the application program receives information other than data, this variable is set to 0. When OPTCD=XBUFBLST is specified, VTAM returns the actual length of the extended buffer list that is built in the buffer list area pointed to by the AREA operand. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.
- X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFHMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFHMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.
NO (B'0')

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

WHATRCV

The field in the RPL extension that indicates what the application program received. It is labeled RPL6WHAT in the RPL extension. The application program should examine the WHATRCV field only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the format shown in Table 1 on page 343.

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>RPL6RCV2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Meaning</td>
</tr>
<tr>
<td>0</td>
<td>DATA</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X'8000'. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a discussion of the meaning of this field.

State changes

See the description of the WHATRCV mask for state changes when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0003'</td>
<td>RECEIVE_SPECIFIC_REJECTED</td>
</tr>
<tr>
<td>RCPI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR_INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR_INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS OR LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0030'</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTEDVECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_OR_SPECIFIED_INCORRECTLY</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC**

**Purpose**
This macroinstruction receives information on a specified conversation. The conversation may be in any continuation mode.

**Usage**
When this macroinstruction is issued, VTAM copies any available data from the conversation that is specified by the CONVID parameter to the data area that is specified by the AREA parameter. The AREALEN parameter specifies the length of the data area. If no data is ready to be received on the conversation, VTAM queues the macroinstruction until data arrives.

When this macroinstruction completes, the RECLEN field indicates how much data was written to the data area. The WHATRCV field indicates what type of data was received.

The application program can issue this macroinstruction when the conversation is in SEND state. In this case, VTAM flushes its SEND buffer, sending all buffered information, along with the SEND indicator, to the partner LU. This changes the conversation to RECEIVE state. VTAM then waits for information to arrive. The remote application program can send data to the local application program after it receives the SEND indication.

If VTAM is processing APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY for a conversation and the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC for the same conversation, the QUALIFY=SPEC request fails with an RCPRI, RCSEC return code of X'0000', X'0003'. (VTAM cannot allow a specific-mode RECEIVE while an any-mode RECEIVE is being processed because if a SEND indication was received on the any-mode RECEIVE while the specific-mode RECEIVE was being processed, a SEND indicator would erroneously be sent to the partner LU as a result of the specific-mode RECEIVE.)

This macroinstruction corresponds to the RECEIVE_AND_WAIT verb described in the LU 6.2 architecture.
Context
For half-duplex conversations, this macroinstruction can be issued from the following conversation states:
- RECEIVE
- SEND
- PEND_END_CONV_LOG
- PEND_RCV_LOG
For full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND/RECEIVE
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- RECEIVE-ONLY_LOG
- PENDING_RESET_LOG
This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
```
APPCCMD  CONTROL  =  RECEIVE  ,  QUALIFY  =  

SPEC 1

,  RPL  =  rpl_address_field
      ( rpl_address_register )

,  AAREA  =  rpl_extension_address_field
      ( rpl_extension_address_register )

,  ACB  =  acb_address_field
      ( acb_address_register )

,  AREA  =  data_area_address_field
      ( data_area_address_register )

,  AREALEN  =  data_area_length
      ( data_area_length_register )

,  BRANCH  =  NO
            YES
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=\texttt{rpl}\_\texttt{extension}\_\texttt{address}\_\texttt{field}
AAREA=\texttt{(rpl}\_\texttt{extension}\_\texttt{address}\_\texttt{register})

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\texttt{acb}\_\texttt{address}\_\texttt{field}
ACB=\texttt{(acb}\_\texttt{address}\_\texttt{register})

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=\texttt{data}\_\texttt{area}\_\texttt{address}\_\texttt{field}
AREA=\texttt{(data}\_\texttt{area}\_\texttt{address}\_\texttt{register})

Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

AREALEN=\texttt{data}\_\texttt{area}\_\texttt{length}
AREALEN=\texttt{(data}\_\texttt{area}\_\texttt{length}\_\texttt{register})

Specifies the length value that is the maximum amount of data the application program is to receive. If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

This field is labeled RPLBUFL in the RPL.
**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CD**

Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.

**CD=DEFER**

Specifies that the conversation state will be in PEND_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CD=IMMED**

Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction.
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**

Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ech_address_field**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**FILL**

Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter corresponds to FILL=LL|BUFFER described in the LU 6.2 architecture. This field is labeled RPL6FILL in the RPL extension.

**FILL=BUFF**

Specifies the application program is to receive data independently of its logical-record format, up to the length specified by the AREALEN field of the RPL. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

**FILL=LL**

Specifies the application program is to receive one logical record, or whatever portion of the logical record is available, up to the length specified by the AREALEN field of the RPL. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT11 field of the RPL.

OPTCD=NBUFFLST
Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. VTAM builds an extended buffer list in the address specified by the AREA parameter. Each entry in the buffer list points to a CSM buffer containing the data being received by the application. The indicator is labeled RPLXBFL and resides within the RPOOPT6 field of the RPL.

Note: Application programs running in TCB-mode supervisor state must specify BRANCH=YES for HPDT requests.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

VTROUTA=vector_address_field
VTROUTA=(vector_address_register)
Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.)

This field is labeled RPL6VAOA in the RPL extension.

VTROUTL=vector_length_field
VTROUTL=(vector_length_register)
Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'
SEND
X'02'
RECEIVE
X'03'
RECEIVE_CONFIRM
X'04'
RECEIVE_CONFIRM_SEND
X'05'
RECEIVE_CONFIRM_DEALLOCATE
X'07'
PENDING_END_CONVERSATION_LOG
X'08'
END_CONVERSATION
For full-duplex conversations, this field can have the following values:

- **X'80'**  
  FDX_RESET

- **X'81'**  
  SEND/RECEIVE

- **X'82'**  
  SEND_ONLY

- **X'83'**  
  RECEIVE_ONLY

- **X'84'**  
  PENDING_SEND/RECEIVE_LOG

- **X'85'**  
  PENDING_RECEIVE-ONLY_LOG

- **X'86'**  
  PENDING_RESET_LOG

**EXPDLLEN**  
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDCRV**  
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**  
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**  
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**  
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**  
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**  
  No FMH-5s are waiting to be received by the application program.

**LOGRCV**  
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

- **YES (B'1')**  
  An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is
the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- **X'0004'**  
  ALLOCATION_ERROR
- **X'0014'**  
  DEALLOCATE_ABEND_PROGRAM
- **X'0018'**  
  DEALLOCATE_ABEND_SERVICE
- **X'001C'**  
  DEALLOCATE_ABEND_TIMER
- **X'0030'**  
  PROGRAM_ERROR_NO_TRUNC
- **X'0034'**  
  PROGRAM_ERROR_PURGING
- **X'0038'**  
  PROGRAM_ERROR_TRUNC
- **X'003C'**  
  SERVICE_ERROR_NO_TRUNC
- **X'0040'**  
  SERVICE_ERROR_PURGING
- **X'0044'**  
  SERVICE_ERROR_TRUNC
- **X'0048'**  
  RESOURCE_FAILURE_NO_RETRY
- **X'005C'**  
  USER_ERROR_CODE_RECEIVED

**NO (B'0')**

Either no error indicator was received or an error indicator was received but indicated that no log data follows.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLLEN**

The field in the RPL that returns to the application program the actual amount of data the application program received. If the application program receives information other than data, this variable is set to 0. When OPTCD=XBUFLST is specified, VTAM returns the actual length of the extended buffer list that is built in the buffer list area pointed to by the AREA operand. This field is labeled RPLRLEN in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.
SENSE
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

WHATRCV
The field in the RPL extension that returns a mask specifying what the application program received. It is labeled RPL6WHAT in the RPL. The application program should examine this WHATRCV mask only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format.
<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning</th>
<th>Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DATA</td>
<td>0</td>
<td>PARTIAL_PS_HEADER</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
<td>1–7</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X’8000’. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a discussion of the meaning of this field.

**State changes**

See the description of the WHATRCV mask for state changes when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0003’</td>
<td>RECEIVE_SPECIFIC_REJECTED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0002’</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0003’</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0004’</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0005’</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0007’</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0008’</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0009’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000A’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000B’</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000D’</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X’0014’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X’0018’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X’001C’</td>
<td>X’0000’</td>
<td>DEALALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X’0024’</td>
<td>X’0000’</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0002’</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0030'</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_OR_SPECIFIED_INCORRECTLY</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOTAUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=REJECT, QUALIFY=CONV**

**Purpose**

This macroinstruction deallocates a conversation abnormally as well as its underlying session when the application program detects a protocol violation on the conversation.

If the conversation is no longer associated with a session when APPCCMD CONTROL=REJECT, QUALIFY=CONV is issued, VTAM does not unbind the session.

**Usage**

When the application program detects a protocol violation on the conversation, it issues this macroinstruction and specifies a sense code on the SENSE parameter. VTAM deallocates the conversation first. If the conversation is still associated with a session, VTAM deactivates the session by issuing an UNBIND of type X'FE', which contains the user-specified sense code. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a list of valid UNBIND sense codes.

As an example, suppose the local application program issues an APPCCMD macroinstruction that completes with a return code of PROGRAM_ERROR_NO_TRUNC and LOGRCV=YES, which indicates that an error is detected and that the partner LU is sending error log data. Also, suppose the local application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to receive the error log data and either no data is received or the data that is received is not error-log data. This means that the partner LU committed a protocol violation, and the application program could issue this macroinstruction to end the conversation and session.

APPCCMD CONTROL=REJECT, QUALIFY=CONV can be issued to cancel an APPCCMD macroinstruction that was issued on the conversation previously. However, it cannot cancel an APPCCMD CONTROL=OPRCNTL, APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY, or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY macroinstruction that has not been matched to a conversation. Nor can it cancel an APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction that was issued previously for the same conversation or an APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL|IALL.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PENDING_DEALLOCATE
- PEND_END_CONV_LOG
- PENDING_SEND
- PEND_RCV_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- RECEIVE_ONLY
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

\[
\text{name APPCCMD CONTROL = REJECT, QUALIFY = } \\
\quad \text{CONV} \\
\quad \text{RPL = rpl\_address\_field} \\
\quad \quad \text{(rpl\_address\_register)} \\
\quad \text{AAREA = rpl\_extension\_address\_field} \\
\quad \quad \text{(rpl\_extension\_address\_register)} \\
\quad \text{ACB = acb\_address\_field} \\
\quad \quad \text{(acb\_address\_register)} \\
\quad \text{BRANCH = NO YES} \\
\quad \text{CONVID = 32-bit\_resource\_id\_field} \\
\quad \quad \text{(32-bit\_resource\_id\_register)} \\
\quad \text{ECB = INTERNAL} \\
\quad \quad \text{ecb\_address\_field} \\
\quad \quad \text{(ecb\_address\_register)} \\
\quad \text{EXIT = exit\_routine\_address\_field} \\
\quad \quad \text{(exit\_routine\_address\_register)} \\
\quad \text{OPTCD = (ASY, SYN, KEEPSRB, NKEEPSRB)} \\
\]
SENSE = 32-bit_unbind_sense_code
( 32-bit_unbind_sense_code_register )

Notes:

1. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRIB is meaningful only for synchronous operations.
8. NKEEPSRIB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=

**AAREA=(**rpl_extension_address_register**)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=

**ACB=(**acb_address_register**)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID**=

**CONVID=(**32-bit_resource_id_register**)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.
ECB=ecb_address_field

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing the APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource. The application program is allowed to issue APPCCMDs against other conversations.

OPTCD=KEEPSRB

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field

RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SENSE=32-bit_unbind_sense_code

SENSE=(32-bit_unbind_sense_code_register)

Indicates the reason for the APPCCMD CONTROL=REJECT macroinstruction. This field specifies a 32-bit UNBIND (X'FE') sense code. VTAM generates an UNBIND (X'FE') carrying the supplied sense code and ends the conversation. This field is labeled RPL6SNSO in the RPL extension. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information on sense codes.)

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following value:

X'08'

END_CONVERSATION

For full-duplex conversations, this field can have the following value:
**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFHM5 in order to receive an FMH-5.

- **NO (B'0')**
  No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFHM5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful processing.

For full-duplex conversations, the conversations state is FDX_RESET after successful processing.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.
### APPCCMD CONTROL=REJECT, QUALIFY=CONVGRP

#### Purpose

This macroinstruction deactivates the session associated with the conversation group and any conversations matched to the session. The application program specifies, through the deactivation type code, that either a protocol violation has occurred or cleanup is necessary.

#### Usage

If the application program detects a protocol violation committed by the partner LU or if the architected processing indicates that a cleanup deactivation of the session should occur, the application program issues APPCCMD CONTROL=REJECT, QUALIFY=CONVGRP to terminate the session. This session can have an active conversation associated with it. If so, the conversation fails with an indication of an abnormal termination.

By using the deactivation type (DEACTYP) parameter, the application program can indicate that VTAM should send either an UNBIND PROTOCOL VIOLATION (X'FE') or an UNBIND CLEANUP (X'0F') to deactivate the session. If the deactivation type parameter is omitted, or is equal to a value other than X'0F' or X'FE', VTAM generates an UNBIND (X'0F'). The sense code parameter is valid only if UNBIND (X'FE') is specified.

The application program must specify the conversation group that is to be deactivated. To do this, it uses the CGID parameters to specify the conversation group identifier.

VTAM posts the application program with successful return codes if no session is active with the specified conversation group identifier.

APPCCMD CONTROL=REJECT, QUALIFY=CONVGRP can be issued without knowledge of any conversations associated with the specified session through the CGID parameter. It corresponds to the DEACTIVATE_CONVERSATION_GROUP verb in the LU 6.2 architecture.
Context
This macroinstruction is not conversation-specific and therefore is not conversation-state-driven.

Syntax

\[
\text{APPCCMD} \quad \text{CONTROL} = \text{REJECT}, \quad \text{QUALIFY} = \quad \text{CONVGRP} \quad \text{RPL} = \quad \text{rpl_address_field} \\
\quad \text{AAREA} = \quad \text{rpl_extension_address_field} \\
\quad \text{ACB} = \quad \text{acb_address_field} \\
\quad \text{BRANCH} = \quad \text{NO}, \quad \text{YES} \\
\quad \text{CGID} = \quad \text{32-bit_conversation_group_id_field} \\
\quad \text{DEACTYP} = \quad \text{8-bit_unbind_type_code} \\
\quad \text{ECB} = \quad \text{INTERNAL} \\
\quad \text{EXIT} = \quad \text{exit_routine_address_field}
\]
Notes:

1. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=\text{rpl\_extension\_address\_field}

AAREA=(\text{rpl\_extension\_address\_register})

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\text{acb\_address\_field}

ACB=(\text{acb\_address\_register})

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.
**CGID=32-bit conversation_group_id_field**

Specifies the 32-bit conversation group ID.

This value can be obtained from a previous APPCCMD CONTROL=ALLOC, CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction. If the CGID operand is not specified, VTAM uses the conversation group ID that is already in the RPL6CGID field on the RPL extension.

The conversation group ID identifies a specific session between two specific LUs. It provides a means by which a VTAM LU 6.2 application program and its partner LU can share serially the same session.

**DEACTYP=8-bit unbind_type_code**

The UNBIND type code can be specified as cleanup (X’0F’) or as protocol violation (X’FE’). If DEACTYP specifies cleanup, the value specified on the SENSE operand will be ignored. However, if DEACTYP specifies protocol error, the UNBIND will flow with the sense code specified by the SENSE operand. If the DEACTYP operand is omitted or a value other than X’0F’ or X’FE’ is entered, VTAM will generate an UNBIND of X’0F’. The application program can be posted with a return code of INVALID_DEACTIVATION_TYPE_CODE, but the session may still have been deactivated successfully. This field is labeled RPL6DETP in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
RPL\text{=}\text{rpl\_address\_field}

RPL\text{=}(\text{rpl\_address\_register})

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SENSE\text{=}32\text{-bit\_unbind\_sense\_code}

SENSE\text{=}((32\text{-bit\_unbind\_sense\_code\_register})

Indicates the reason for the APPCCMD CONTROL\text{=}REJECT macroinstruction. This field specifies a 32-bit UNBIND (X'FE') sense code. VTAM generates an UNBIND (X'FE') carrying the supplied sense code and ends the conversation. This field is labeled RPL6SNSO in the RPL extension. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information on sense codes.)

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

FDB2

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV\text{=}YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL\text{=}RCVFMH5 in order to receive an FMH-5.

NO (B'0')

No FMH-5s are waiting to be received by the application program.

RCPRI

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD\text{=}X'00' and FDB2\text{=}X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD\text{=}X'00' and FDB2\text{=}X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD

The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL\text{=}ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL\text{=}RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.
State changes
Conversation states do not apply to this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0020'</td>
<td>PARAMETER_ERROR—PREVIOUS_REJECT_REQUEST_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0027'</td>
<td>PARAMETER_ERROR—INVALID_DEACTIVATION_TYPE_CODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002A'</td>
<td>PARAMETER_ERROR—INVALID_CGID_VALUE_SPECIFIED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=REJECT, QUALIFY=SESSION

Purpose
This macroinstruction deactivates the session and any conversation matched to this session. The application program specifies, through the deactivation type code, that either a protocol violation has occurred or cleanup is necessary.

Usage
If the application program detects a protocol violation committed by the partner LU or if the architected processing indicates that a cleanup deactivation of the session should occur, the application program issues APPCCMD CONTROL=REJECT, QUALIFY=SESSION to terminate the session. This session can have an active conversation associated with it. If so, the conversation fails with an indication of an abnormal termination. The application must issue an APPCCMD to receive the conversation failure notification and cause conversation cleanup.

By using the deactivation type (DEACTYP) parameter, the application program can indicate that VTAM should send either an UNBIND_PROTOCOL_VIOLATION (X'FE') or an UNBIND_CLEANUP (X'0F') to deactivate the session. If the deactivation type parameter is omitted, or is equal to a value other than X'0F' or X'FE', VTAM generates an UNBIND (X'0F'). The sense code parameter is ignored unless an UNBIND (X'FE') is specified.
The application program must specify the session that is to be deactivated. To do this, it uses the SESSID and SESSIDL parameters to specify the session instance identifier. These parameters were made available to the conversation at conversation allocation from the APPCCMD CONTROL=RCVFMH5 macroinstruction and the APPCCMD CONTROL=ALLOC macroinstruction.

VTAM posts the application program with successful return codes if no session is active with the specified session identifier and session identifier length.

APPCCMD CONTROL=REJECT, QUALIFY=SESSION can be issued without knowledge of any conversations associated with the specified session through the SESSID parameter.

**Context**

This macroinstruction is not conversation-specific and therefore is not conversation-state-driven.

**Syntax**

```
APPCCMD — — CONTROL — = — REJECT — , — QUALIFY — = — SESSION

, — RPL — = — rpl_address_field

, — AAREA — = — rpl_extension_address_field

, — ACB — = — acb_address_field

, — BRANCH — = — NO — YES

, — DEACTYP — = — 8-bit_unbind_type_code

, — ECB — = — INTERNAL

, — EXIT — = — exit_routine_address_field
```
Input parameters
The following information shows descriptions of the input parameters:

**AAREA=**rpl_extension_address_field

**AAREA=(rpl_extension_address_register)**

Specifications the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**acb_address_field

**ACB=(acb_address_register)**

Specifications the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifications whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use
BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**DEACTYP=8-bit_unbind_type_code**

DEACTYP=(8-bit_unbind_type_code_register)
The UNBIND type code can be specified as cleanup (X'0F') or as protocol violation (X'FE'). If DEACTYP specifies cleanup, the value specified on the SENSE operand will be ignored. However, if DEACTYP specifies protocol error, the UNBIND will flow with the sense code specified by the SENSE operand. If the DEACTYP operand is omitted or a value other than X'0F' or X'FE' is entered, VTAM will generate an UNBIND of X'0F'. The application program can be posted with a return code of INVALID_DEACTIVATION_TYPE_CODE, but the session may still have been deactivated successfully. This field is labeled RPL6DETP in the RPL extension.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
RPL=rpl_address_field
RPL=(rpl_address_register)
   Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SENSE=32-bit.unbind_sense_code
SENSE=(32-bit.unbind_sense_code_register)
   Indicates the reason for the APPCCMD CONTROL=REJECT macroinstruction. This field specifies a 32-bit UNBIND (X'FE') sense code. VTAM generates an UNBIND (X'FE') carrying the supplied sense code and ends the conversation. This field is labeled RPL6NSNO in the RPL extension. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information on sense codes.)

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
   Specifies the session to be deactivated. The session instance identifier must refer to an active session. (A session must be activated before it can be deactivated.) The session instance identifier is passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. A session that is in pending activation state cannot be specified. A conversation that is matched to this session fails with a session outage notification. This field is labeled RPL6SSID in the RPL extension.

SESSIDL=session_instance_id_length
SESSIDL=(session_instance_id_length_register)
   Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

FDB2
   The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
   The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
   The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

   YES (B'1')
      One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

   NO (B'0')
      No FMH-5s are waiting to be received by the application program.

RCPRI
   The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
   The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is
labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**State changes**

Conversation states do not apply to this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0020'</td>
<td>PARAMETER_ERROR—PREVIOUS_REJECT_REQUEST_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>PARAMETER_ERROR—INVALID_SESSION_INSTANCE_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0027'</td>
<td>PARAMETER_ERROR—INVALID_DEACTIVATION_TYPE_CODE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=RESETRCV**

**Purpose**

This macroinstruction changes the existing continuation modes of a conversation. For example, it can change the conversation from continue-specific (CS) mode to logical-record-continue-any (LLCA) mode for receiving normal information.

This macroinstruction can also change the existing mode for receiving expedited information.

**Usage**

When this macroinstruction is issued, VTAM changes the continuation mode for receiving normal information of the conversation specified with the CONVID parameter to the continuation mode specified on the CONMODE parameter.
VTAM also changes the expedited information mode of the conversation specified with the CONXMOD parameter to the expedited information mode specified on the CONXMOD parameter.

For a complete discussion of continuation modes and an example of how this macroinstruction can be used, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PEND_END_CONV_LOG
- PEND_RCV_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```plaintext
name APPCCMD — CONTROL = RESETRCV 
, QUALIFY = NULL 
, RPL = rpl_address_field 
( rpl_address_register ) 
, AAREA = rpl_extension_address_field 
( rpl_extension_address_register ) 
, ACB = acb_address_field 
( acb_address_register ) 
, BRANCH = NO YES 2
```
Notes:

1 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:
AAREA=\textit{rpl\_extension\_address\_field}  
\textbf{AAREA=(rpl\_extension\_address\_register)}  
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\textit{acb\_address\_field}  
\textbf{ACB=(acb\_address\_register)}  
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

\textbf{BRANCH}  
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

\textbf{BRANCH=NO}  
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

\textbf{BRANCH=YES}  
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

\textbf{CONMODE}  
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

\textbf{CONMODE=BUFFCA}  
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

\textbf{CONMODE=CS}  
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

\textbf{CONMODE=LLCA}  
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

\textbf{CONMODE=SAME}  
Specifies that the continuation mode of the conversation is to remain unchanged.

\textbf{CONVID=32-bit\_resource\_id\_field}  
\textbf{CONVID=(32-bit\_resource\_id\_register)}  
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

\textbf{CONXMOD}  
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.
CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited
information can be received by either a specific-type macroinstruction or an any-type
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD
CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited
information can be received only by a specific-type macroinstruction, such as, APPCCMD
CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the
completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the
completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single
APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD
completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be
any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD
completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the
function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the
RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the
application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLEXTDS field of
the RPL.

When the application program regains control after issuing this APPCCMD asynchronously, it is
prevented from issuing another APPCCMD against the same conversation resource (that
processes on the SEND/RECEIVE queue if the conversation is half-duplex, or on the SEND queue if
the conversation is full-duplex) until the command has completed. The exceptions to this are the
APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction and the abnormal termination
APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstruction. The application program can issue
APPCCMDs against the same conversation resource that processes on the RECEIVE (if the
conversation is full-duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For
more information about conversation queues, refer to z/OS Communications Server: SNA
Programmer’s LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY
is recommended when issuing the APPCCMD on a full-duplex conversation.
**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**
**RPL=(rpl_address_register)**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**
The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**
The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'
SEND

X'02'
RECEIVE

X'03'
RECEIVE_CONFIRM

X'04'
RECEIVE_CONFIRM_SEND

X'05'
RECEIVE_CONFIRM_DEALLOCATE

X'07'
PENDING_END_CONVERSATION_LOG

X'08'
END_CONVERSATION

X'09'
PENDING_SEND

X'0A'
PENDING_RECEIVE_LOG

For full-duplex conversations, this field can contain the following values:

X'80'
FDX_RESET

X'81'
SEND/RECEIVE

X'82'
SEND_ONLY

X'83'
RECEIVE_ONLY

X'84'
PENDING_SEND/RECEIVE_LOG
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B’1’)
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B’0’)
No FMH-5s are waiting to be received by the application program.

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

There are no state changes associated with this macroinstruction.

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.
<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPCCAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SEND, QUALIFY=CONFIRM**

**Purpose**

This macroinstruction sends a confirmation request on a half-duplex conversation to a remote application program and waits for a confirmation reply (either synchronously or asynchronously).

**Usage**

This macroinstruction can be used only for half-duplex conversations.

When this macroinstruction is issued, VTAM flushes the SEND buffer of the specified conversation and sends a confirmation request. This macroinstruction completes only after the partner LU receives the confirmation request and issues APPCCMD CONTROL=SEND, QUALIFY=CONFRMD.

This macroinstruction enables the local and remote application programs to synchronize their processing with one another. The application program can use this APPCCMD for various transaction program-level functions. For example:

- The application program can issue this APPCCMD immediately after an APPCCMD CONTROL=ALLOC macroinstruction in order to determine whether the allocation of the conversation is successful before sending any data.
- The application program can issue this APPCCMD as a request for acknowledgment of data that it sent to the remote program.

The application program must ensure that APPCCMD CONTROL=SEND, QUALIFY=CONFIRM is not issued by a transaction program against a conversation that was allocated with a synchronization level of NONE.

This macroinstruction corresponds to the CONFIRM verb described in the LU 6.2 architecture.
Context
For half-duplex conversations, this macroinstruction can be issued from following conversation states:
• SEND
• PENDING_SEND
This macroinstruction is not allowed on full-duplex conversations.
This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
name
APPCCMD          CONTROL      = SEND
                QUALIFY    =  
                CONFIRM 1

, RPL = rpl_address_field
    ( rpl_address_register )

, AAREA = rpl_extension_address_field
    ( rpl_extension_address_register )

, ACB = acb_address_field
    ( acb_address_register )

, BRANCH = NO 3
            YES

, CONMODE = BUFFCA 1
    CS
    LLCA
    SAME

, CONVID = 32-bit_resource_id_field
    ( 32-bit_resource_id_register )

, CONXMOD = CA 1
    CS
    SAME
```
Input parameters

The following information shows descriptions of the input parameters:

**AAREA=**`rpl_extension_address_field`
**AAREA=(rpl_extension_address_register)**
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**`acb_address_field`
**ACB=(acb_address_register)**
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
**BRANCH=NO**
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH= YES**
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- **CONMODE=BUFFCA**
  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

- **CONMODE=CS**
  Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data.

- **CONMODE=LLCA**
  Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

- **CONMODE=SAME**
  Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**
**CONVID=(32-bit_resource_id_register)**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

- **CONXMOD=CA**
  Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

- **CONXMOD=CS**
  Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

- **CONXMOD=SAME**
  Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.
ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

This field can have the following values:
X'01' SEND
X'02' RECEIVE
X'03' RECEIVE_CONFIRM
X'04' RECEIVE_CONFIRM_SEND
X'05'
   RECEIVE_CONFIRM_DEALLOCATE
X'07'
   PENDING_END_CONVERSATION_LOG
X'08'
   END_CONVERSATION
X'09'
   PENDING_SEND
X'0A'
   PENDING_RECEIVE_LOG

EXPDLEN
   The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has
   meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
   The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is
   labeled RPL6EXDR in the RPL extension.

FDB2
   The field in the RPL in which a global VTAM secondary return code is returned to the application
   program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
   The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the
   application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of
   the longest FMH-5 to be received by the application program. This field has meaning only when
   FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
   The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The
   indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL
   extension.

   YES (B'1')
      One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set
      to YES as long as an FMH-5 is waiting to be received by the application program. The application
      program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

   NO (B'0')
      No FMH-5s are waiting to be received by the application program.

LOGRCV
   The field in the RPL extension that returns an indication of whether error log data is expected. The
   indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL
   extension.

   YES (B'1')
      An FMH-7 was received that specified that error log data follows. The application program must
      issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is
      the responsibility of the application program to perform an optional receive check after issuing
      APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log
      data was sent by the partner LU. The data must be error log data and it must be in the form of a
      GDS variable.

      LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

      X'0004'
         ALLOCATION_ERROR
      X'0014'
         DEALLOCATE_ABEND_PROGRAM
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

- X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RSRTRN=EXPD.
SIGRCV

The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPI extension.

**YES (B'1')**

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes

There are no state changes associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (REMOTE PROGRAM REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=SEND, QUALIFY=CONFRMD**

**Purpose**
This macroinstruction sends a positive confirmation reply to the remote application program on a half-duplex conversation.

**Usage**
This macroinstruction can only be used for half-duplex conversations.
When the application program receives a CONFIRM indication in the WHATRCV field after an APPCCMD CONTROL=RECEIVE macroinstruction, the application issues this macroinstruction to indicate that all the data that was sent by the CONFIRM indication has been received and is acceptable. This allows an application program to synchronize processing with its partner application.

If the application program receives a CONFIRM indication and it detects an error in the data it received before the CONFIRM, it can issue APPCCMD CONTROL=SEND, QUALIFY=ERROR to send a negative reply to the CONFIRM.

This macroinstruction corresponds to the CONFIRMED verb described in the LU 6.2 architecture.

**Context**

This macroinstruction may only be issued from the following conversation states on a half-duplex conversation:

- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE

This macroinstruction is not allowed on a full-duplex conversation.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```plaintext
APPCCMD  CONTROL  =  SEND  ,  QUALIFY  =  CONFRMD
,  RPL  =  rpl_address_field
,  AAREA  =  rpl_extension_address_field
,  ACB  =  acb_address_field
,  BRANCH  =  NO  YES
,  CONMODE  =  BUFFCA  CS  LLCA  SAME
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=**rpl_extension_address_field
**AAREA=(**rpl_extension_address_register**)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**acb_address_field
**ACB=(**acb_address_register**)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs
using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.
CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(epl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

This field can have the following values:

X'01'
SEND
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMHSLEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMHSLEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5
macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
These changes are applicable when RCPRI indicates OK.

RECEIVE state is entered when an indicator of CONFIRM, DATA_CONFIRM, or DATA_COMPLETE_CONFIRM was received on the preceding APPCCMD CONTROL=RECEIVE.

SEND state is entered when an indicator of CONFIRM_SEND, DATA_CONFIRM_SEND, or DATA_COMPLETE_CONFIRM_SEND was received on the preceding APPCCMD CONTROL=RECEIVE.

END_CONV state is entered when an indicator of CONFIRM_DEALLOCATE, DATA_CONFIRM_DEALLOCATE, or DATA_COMPLETE_CONFIRM_DEALLOCATE was received on the preceding APPCCMD CONTROL=RECEIVE.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (PARTNER LU REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPCCCAPEABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED—CONTROL/QUALIFY_VALUE_INVALID_FOR_FULL-DUPLEX_CONVERSATION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=SEND, QUALIFY=DATA**

**Purpose**
This macroinstruction sends data to a partner LU.

**Usage**
This macroinstruction transfers data that is specified by the AREA parameter into the SEND buffer of the conversation that is specified by the CONVID parameter. When there is more data in the conversation's SEND buffer than the maximum RU size for the conversation's session, an RU is sent to the partner LU. If the data does not exceed a maximum RU size, the data in the buffer remains there until the application program sends more data or causes the SEND buffer to be flushed.

**Note:** If OPTCD=XBUFLST is specified on this macroinstruction, all of the data is sent to the partner LU, even if the data does not exceed the maximum RU size.

The AREA parameter can specify a single data area to be sent, or it can specify a buffer list that points to multiple data areas to be sent. The OPTCD parameter specifies which of these methods is used.

For a complete discussion of sending data, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

This macroinstruction corresponds to the SEND_DATA verb described in the LU 6.2 architecture.

**Context**
For half-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND
- PENDING_SEND

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
name APPCCMD CONTROL=SEND, QUALIFY=DATA

, RPL = rpl_address_field
  ( rpl_address_register )

, AAREA = rpl_extension_address_field
  ( rpl_extension_address_register )
```
ACB = acb_address_field

AREA = data_area_or_buffer_list_address_field

BRANCH = NO

CONMODE = BUFFCA
- CS
- LLCA
- SAME

CONVID = 32-bit_resource_id_field

CONXMOD = CA
- CS
- SAME

CRYPT = NO

ECB = INTERNAL
- ecb_address_field

EXIT = exit_routine_address_field
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA = rpl_extension_address_field
AAREA = (rpl_extension_address_register)
   Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB = acb_address_field
ACB = (acb_address_register)
   Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

AREA = data_area_or_buffer_list_address_field
AREA = (data_area_or_buffer_list_address_register)
   Specifies the address of a data buffer or buffer list.
   - If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
• If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

• If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST is specified on this macroinstruction, VTAM performs an internal flush of any data remaining in the send buffer, even if it does not exceed the maximum RU size.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.
CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ech_address_field
ECB=(ech_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

OPTCD=XBUFFLST
 Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:
Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.

Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X'002C', X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

RECLEN=\texttt{data\_length}

\texttt{RECLEN=(data\_length\_register)}

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

RPL=\texttt{rpl\_address\_field}

\texttt{RPL=(rpl\_address\_register)}

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

\textbf{RPL and RPL extension fields modified by macroinstruction}

The following information shows descriptions of RPL and RPL extension fields:

\textbf{CONSTATE}

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can contain the following values:

- \texttt{X'01'} \hspace{1cm} SEND
- \texttt{X'02'} \hspace{1cm} RECEIVE
- \texttt{X'03'} \hspace{1cm} RECEIVE\_CONFIRM
- \texttt{X'04'} \hspace{1cm} RECEIVE\_CONFIRM\_SEND
- \texttt{X'05'} \hspace{1cm} RECEIVE\_CONFIRM\_DEALLOCATE
- \texttt{X'07'} \hspace{1cm} PENDING\_END\_CONVERSATION\_LOG
- \texttt{X'08'} \hspace{1cm} END\_CONVERSATION
- \texttt{X'09'} \hspace{1cm} PENDING\_SEND
- \texttt{X'0A'} \hspace{1cm} PENDING\_RECEIVE\_LOG

For full-duplex conversations, this field can have the following values:

- \texttt{X'80'} \hspace{1cm} FDX\_RESET
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

Note: The LOGRCV field is reserved if this macroinstruction is issued on a full-duplex conversation.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. The application program must perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable. LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
ALLOCATION_ERROR
X'0014'
DEALLOCATE_ABEND_PROGRAM
X'0018'
DEALLOCATE_ABEND_SERVICE
X'001C'
DEALLOCATE_ABEND_TIMER
X'0030'
PROGRAM_ERROR_NO_TRUNC
X'0034'
PROGRAM_ERROR_PURGING
X'0038'
PROGRAM_ERROR_TRUNC
X'003C'
SERVICE_ERROR_NO_TRUNC
X'0040'
SERVICE_ERROR_PURGING
X'0044'
SERVICE_ERROR_TRUNC
X'005C'
USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.
SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
No state changes are associated with this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.
<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td></td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td></td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td></td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td></td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td></td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td></td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td></td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td></td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td></td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td></td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td></td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=SEND, QUALIFY=DATACON

Purpose

This macroinstruction sends data that is supplied by the application program and any data that is already in the SEND buffer to a partner application program on a half-duplex conversation. The data is followed by a confirmation request.

Usage

This macroinstruction can only be used on a half-duplex conversation.

VTAM places the data specified by the AREA parameter in the SEND buffer of the conversation specified by the CONVID parameter. VTAM sends all data in the SEND buffer to the partner LU. The data is followed by a confirmation request. This macroinstruction completes only after a confirmation reply is received from the partner LU. The application program must ensure that the data that it sends completes a logical record.

For more information on sending and responding to confirmation requests, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

This macroinstruction corresponds to the SEND_DATA followed by CONFIRM verbs described in the LU 6.2 architecture.
Context
This macroinstruction can be issued on a half-duplex conversation from the following conversation states:
• SEND
• PENDING_SEND
This macroinstruction is not allowed on a full-duplex conversation.
This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
name -- APPCCMD -- CONTROL -- = -- SEND -- , -- QUALIFY -- = --

DATACON 1

, -- RPL = rpl_address_field
( rpl_address_register )

, -- AAREA = rpl_extension_address_field
( rpl_extension_address_register )

, -- ACB = acb_address_field
( acb_address_register )

, -- AREA = data_area_or_buffer_list_address_field
( data_area_or_buffer_list_address_register )

, -- BRANCH = NO
YES

, -- CONMODE = BUFFCA
CS
LLCA
SAME

, -- CONVID = 32-bit_resource_id_field
( 32-bit_resource_id_register )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. KEEPSRB is meaningful only for synchronous operations.
7. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:
AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
    Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
    Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=data_area_or_buffer_list_address_field
AREA=(data_area_or_buffer_list_address_register)
    Specifies the address of a data buffer or buffer list.
    • If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
    • If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
    • If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.
        If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.
        This field is labeled RPLAREA in the RPL.

BRANCH
    Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

    BRANCH=NO
        Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

    BRANCH=YES
        Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
    Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

    CONMODE=BUFFCA
        Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the
application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**
**CONVID=(32-bit_resource_id_register)**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**CRYPT**
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

**CRYPT=NO**
Do not encrypt data before it is sent.

**CRYPT=YES**
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

**Note:** If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCCMD macroinstruction. The indicator resides within the RPOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCCMD macroinstruction completes.
ECB=ecb_address_field
ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing the APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the SEND/RECEIVE queue until the command has completed. The exception to this is the APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction. The application can issue APPCCMDs against the same conversation resource that processes on the EXPEDITED SEND, EXPEDITED RECEIVE and TESTSTAT queues. For more information about conversation queues refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide

The application program is allowed to issue APPCCMDs against other conversations.

OPTCD=KEEPSRB

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFFLST

Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST

Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.
OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X'002C', X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

RECLEN=\textit{data\_length} 

\textbf{RECLEN=}(\textit{data\_length\_register})

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

RPL=\textit{rpl\_address\_field} 

\textbf{RPL=(rpl\_address\_register)}

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

\textbf{CONSTATE}

The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

This field can contain the following values:

\textbf{X'01'}

\textbf{SEND}

\textbf{X'02'}

\textbf{RECEIVE}

\textbf{X'03'}

\textbf{RECEIVE\_CONFIRM}

\textbf{X'04'}

\textbf{RECEIVE\_CONFIRM\_SEND}

\textbf{X'05'}

\textbf{RECEIVE\_CONFIRM\_DEALLOCATE}

\textbf{X'07'}

\textbf{PENDING\_END\_CONVERSATION\_LOG}
EXPDLLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
ALLOCATION_ERROR

X'0014'
DEALLOCATE_ABEND_PROGRAM

X'0018'
DEALLOCATE_ABEND_SERVICE

X'001C'
DEALLOCATE_ABEND_TIMER
X'0030'
  PROGRAM_ERROR_NO_TRUNC
X'0034'
  PROGRAM_ERROR_PURGING
X'0038'
  PROGRAM_ERROR_TRUNC
X'003C'
  SERVICE_ERROR_NO_TRUNC
X'0040'
  SERVICE_ERROR_PURGING
X'0044'
  SERVICE_ERROR_TRUNC
X'005C'
  USER_ERROR_CODE_RECEIVED

NO (B'0')
  Either no error indicator was received or an error indicator was received but indicated that no log
data follows.

RCPRI
  The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the
application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is
labeled RPL6RCPR in the RPL extension.

RCSEC
  The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the
application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is
labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the
result of the macroinstruction processing.

RPLXSRV
  A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT
request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL,
the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and
not posted complete. It is the application’s responsibility to examine the RPLXSRV bit and determine
if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS
Communications Server: SNA Programmer’s LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
  The field in the RPL in which a global VTAM primary return code is returned to the application
program. This field is labeled RPLRTNCD in the RPL.

SENSE
  The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL
extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also
can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the
session for the conversation was deactivated. Not all RCPRI values have sense data associated with
them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an
FMH-7 sense code that was not interpreted by VTAM.

SIGDATA
  The field in the RPL extension in which the signal code and signal extension fields of a received
SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES.
This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote
application program.
**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**

The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

**YES (B'1')**

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**STSHBF**

The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**

The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (REMOTE PROGRAM REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>RCPI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0004'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0005'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0006'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0007'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0009'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=SEND, QUALIFY=DATAFLU

**Purpose**

This macroinstruction sends data supplied by the application program as well as any data that is already in the SEND buffer to the partner application.

**Usage**

This macroinstruction combines the functions of two macroinstructions: APPCCMD CONTROL=SEND, QUALIFY=DATA followed by APPCCMD CONTROL=SEND, QUALIFY=FLUSH. VTAM places the data that is specified by the AREA parameter in the SEND buffer of the conversation that is specified by the CONVID parameter. VTAM sends all data in the SEND buffer to the partner LU.

This macroinstruction corresponds to SEND_DATA followed by FLUSH verbs described in the LU 6.2 architecture.


**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- PENDING_SEND

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD CONTROL=SEND, QUALIFY=DATAFLU
```
LU 6.2 macroinstruction syntax and operands
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPXR is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** rpl_extension_address_field

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** acb_address_field

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with
transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=data_area_or_buffer_list_address_field**

**AREA=(data_area_or_buffer_list_address_register)**

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

  If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**
  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**
  
  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- **CONMODE=BUFFCA**
  
  Indicates that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

- **CONMODE=CS**
  
  Indicates that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.
CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.
**EXIT=exit_routine_address_field**
**EXIT=(exit_routine_address_register)**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=BUFFLST**
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLen field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

**OPTCD=NBUFLST**
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLen field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.
OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X'002C', X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

RECLEN=\texttt{data\_length}

RECLEN=\texttt{(data\_length\_register)}

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

RPL=\texttt{rpl\_address\_field}

RPL=\texttt{(rpl\_address\_register)}

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

\textbf{CONSTATE}

The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- \texttt{X'01'}
  - \texttt{SEND}
- \texttt{X'02'}
  - \texttt{RECEIVE}
- \texttt{X'03'}
  - \texttt{RECEIVE\_CONFIRM}
- \texttt{X'04'}
  - \texttt{RECEIVE\_CONFIRM\_SEND}
- \texttt{X'05'}
  - \texttt{RECEIVE\_CONFIRM\_DEALLOCATE}
- \texttt{X'07'}
  - \texttt{PENDING\_END\_CONVERSATION\_LOG}
For full-duplex conversations, this field can have the following values:

- **FDX_RESET**
- **SEND/RECEIVE**
- **SEND_ONLY**
- **RECEIVE_ONLY**
- **PENDING_SEND/RECEIVE_LOG**
- **PENDING_RECEIVE-ONLY_LOG**
- **PENDING_RESET_LOG**

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B’1’)**
  
  One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B’0’)**
  
  No FMH-5s are waiting to be received by the application program.

**LOGRCV**

The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**Note:** The LOGRCV field is reserved if this macroinstruction is issued on a full-duplex conversation.
YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. The application program must perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable.
LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
  ALLOCATION_ERROR
X'0014'
  DEALLOCATE_ABEND_PROGRAM
X'0018'
  DEALLOCATE_ABEND_SERVICE
X'001C'
  DEALLOCATE_ABEND_TIMER
X'0030'
  PROGRAM_ERROR_NO_TRUNC
X'0034'
  PROGRAM_ERROR_PURGING
X'0038'
  PROGRAM_ERROR_TRUNC
X'003C'
  SERVICE_ERROR_NO_TRUNC
X'0040'
  SERVICE_ERROR_PURGING
X'0044'
  SERVICE_ERROR_TRUNC
X'005C'
  USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.
The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.
RTNCD

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE

The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

SIGDATA

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV

The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B’1’)

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF

The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS

The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5...
macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes

No state changes are associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002C'</td>
<td>PARAMETER_ERROR—INVALID_EXPEDITED_DATA_LENGTH</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABDOTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOTAUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_TIME</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>ERROR_INDICATION_RECEIVED—ALLOCATION_ERROR</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>ERROR_INDICATION_RECEIVED—UNKNOWN_ERROR_CODE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0002'</td>
<td>CSM_DETECTED_ERROR—INVALID_BUFFER_TOKEN_SPECIFIED</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0003'</td>
<td>CSM_DETECTED_ERROR—INVALID_INSTANCE_ID_SPECIFIED</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SEND, QUALIFY=ERROR**

**Purpose**

This macroinstruction informs the partner LU that the local application program detects an error.
Usage

When this macroinstruction is issued, VTAM builds an FMH-7, based on the TYPE and SENSE parameters, to represent the error that the application program detected.

The application program can specify one of the following types of errors:

• PROGRAM—error in an end-user transaction program
• SERVICE—error in a service component of a transaction program
• USER—user-specified error.

VTAM determines the sense code to place in the FMH-7 for program and service errors. The application program specifies the sense code on the SENSE parameter for user errors. The sense code specified must be appropriate to the error. Otherwise, improper processing of the macroinstruction might result. For a list of valid sense codes for an FMH-7, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

A negative response must be sent to the partner LU before the FMH-7 can be transmitted if the conversation is in one of the following states:

• RECEIVE
• PEND_SEND
• RECEIVE_CONFIRM
• RECEIVE_CONFIRM_SEND
• RECEIVE_CONFIRM_DEALLOCATE

VTAM flushes the SEND buffer before the FMH-7 is created and a negative response is not sent if the conversation is in one of the following states:

• SEND
• SEND/RECEIVE
• SEND_ONLY
• PENDING_SEND/RECEIVE_LOG

For half-duplex conversations, the FMH-7 (and error log data that is supplied) is not sent to the partner LU until the application program issues a macroinstruction such as APPCCMD CONTROL=SEND, QUALIFY=FLUSH that causes the SEND buffer to be flushed. For full-duplex conversations, the FMH-7 is sent immediately to the conversation partner.

This macroinstruction corresponds to the SEND_ERROR verb described in the LU 6.2 architecture.

For more details on error handling, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

• SEND
• RECEIVE
• RECEIVE_CONFIRM
• RECEIVE_CONFIRM_SEND
• RECEIVE_CONFIRM_DEALLOCATE
• PENDING_RECEIVE_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

• SEND/RECEIVE
• SEND_ONLY
• PENDING_SEND/RECEIVE_LOG
This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
name --  APPCCMD --  CONTROL = --  SEND -- ,  QUALIFY = --

ERROR ^1

, --  RPL = --  rpl_address_field
     ( --  rpl_address_register -- )

, --  AAREA = --  rpl_extension_address_field
     ( --  rpl_extension_address_register -- )

, --  ACB = --  acb_address_field
     ( --  acb_address_register -- )

, --  AREA = --  optional_log_data_area_address_field
     ( --  optional_log_data_area_address_register -- )

, --  BRANCH = --  NO
     YES

, --  CONMODE = --  BUFFCA
     CS
     LLCA
     SAME

, --  CONVID = --  32-bit_resource_id_field
     ( --  32-bit_resource_id_register -- )

, --  CONXMOD = --  CA
     CS
     SAME
```
Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.
**ACB=acb_address_field**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=optional_log_data_area_address_field**

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner LU. The application program is responsible for placing the error log data into the local system log. VTAM treats the error log GDS variable the same as other conversation data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.
CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLEXTDS field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.
The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=data_length**
**RECLEN=(data_length_register)**
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A 0 value in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

**RPL=rpl_address_field**
**RPL=(rpl_address_register)**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**SENSE=user-supplied_32-bit_fmh-7_sense_code**
**SENSE=(user-supplied_32-bit_fmh-7_sense_code_register)**
Specifies the sense code that VTAM places in the FMH-7. This field is applicable only when TYPE=USER is specified. It is labeled RPL6SNSO in the RPL extension.

**TYPE**
Specifies the type of error being reported. This field is intended to distinguish between errors to be reported to end-user transaction programs and errors to be reported to a service component, such as a mapped conversation component, of the LU. This field is labeled RPL6TYPE in the RPL extension.

**TYPE=PROGRAM**
Specifies that an end-user transaction program error is being reported. VTAM determines the appropriate sense code to be placed in the FMH-7 based upon the state of the conversation and of the LU's SEND buffer. VTAM also determines whether the FMH-7 should be preceded by a negative response, based upon the current state of the conversation.

VTAM will place a sense code of either X'08890000' or X'08890001' in the FMH-7 for this type of error.

**TYPE=SERVICE**
Specifies that a service-component error is being reported. VTAM determines the appropriate sense code to be placed in the FMH-7 based upon the state of the conversation and of the LU's SEND buffer. VTAM also determines whether the FMH-7 should be preceded by a negative response, based upon the current state of the conversation.

VTAM will place a sense code of either X'08890100' or X'08890101' in the FMH-7 for this type of error.

**TYPE=USER**
Specifies that the application program is providing to VTAM a user-specified sense code that is to be placed in the FMH-7. The FMH-7 sense code is passed to VTAM through the SENSE field of the RPL extension. It is the responsibility of the application program to supply a valid FMH-7 sense code. This user-specified sense code must be appropriate for the error. Otherwise, improper processing of the macroinstruction might occur. VTAM determines whether the FMH-7 should be preceded by a negative response, based upon the current state of the conversation. For a list of sense codes that the application program can use, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- **X'01'** SEND
- **X'02'** RECEIVE
- **X'03'** RECEIVE_CONFIRM
- **X'04'** RECEIIVE_CONFIRM_SEND
- **X'05'** RECEIVE_CONFIRM_DEALLOCATE
- **X'07'** PENDING_END_CONVERSATION_LOG
- **X'08'** END_CONVERSATION
- **X'09'** PENDING_SEND
- **X'0A'** PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- **X'80'** FDX_RESET
- **X'81'** SEND/RECEIVE
- **X'82'** SEND_ONLY
- **X'83'** RECEIVE_ONLY
- **X'84'** PENDING_SEND/RECEIVE_LOG
- **X'85'** PENDING_RECEIVE-ONLY_LOG
- **X'86'** PENDING_RESET_LOG

**EXPDLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

Note: The LOGRCV field is reserved if this macroinstruction is issued on a full-duplex conversation.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. The application program must perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable. LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
ALLOCATION_ERROR

X'0014'
DEALLOCATE_ABEND_PROGRAM

X'0018'
DEALLOCATE_ABEND_SERVICE

X'001C'
DEALLOCATE_ABEND_TIMER

X'0030'
PROGRAM_ERROR_NO_TRUNC

X'0034'
PROGRAM_ERROR_PURGING

X'0038'
PROGRAM_ERROR_TRUNC

X'003C'
SERVICE_ERROR_NO_TRUNC

X'0040'
SERVICE_ERROR_PURGING

X'0044'
SERVICE_ERROR_TRUNC
X'005C'
USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. If the session for the conversation was deactivated, this code explains why. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.
- Hex 00010001 indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer
list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
These changes are applicable when RCPRI indicates OK.
For half-duplex conversations, the conversation state is SEND after successful completion.
For full-duplex conversations, no conversation state changes occur.
See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes. The return codes that can be returned depend on the state of the conversation at the time this APPCCMD is issued.

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in SEND state, the following values can be returned:

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>RCPI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_ORRESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in SEND/RECEIVE, SEND_ONLY, or PENDING_SEND/RECEIVE_log state, the following values can be returned:

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_TIME</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>ERROR_INDICATION_RECEIVED—ALLOCATION_ERROR</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>ERROR_INDICATION_RECEIVED—UNKNOWN_ERROR_CODE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_NO_RETRY</td>
</tr>
</tbody>
</table>

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in RECEIVE, PEND_SEND, or PEND_RCV_LOG state, the following values can be returned:

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0080'</td>
<td>X'0000'</td>
<td>DEALLOCATE_NORMAL</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT_ REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in RECEIVE_CONFIRM, RECEIVE_CONFIRM_SEND, or RECEIVE_CONFIRM_DEALLOCATE state, the following values can be returned:

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO ECB FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST INVALID FOR ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL BLOCK INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID DATA ADDRESS OR LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS MACROINSTRUCTION OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED FOR NON-APPCC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE OR RESOURCE SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SEND, QUALIFY=FLUSH**

**Purpose**
This macroinstruction flushes the VTAM SEND buffer associated with the specified conversation.

**Usage**
This macroinstruction is useful for optimizing processing between the application program and its partner LU. VTAM normally buffers the data from consecutive SEND macroinstructions until it has enough data for transmission. With this macroinstruction, the application program causes VTAM to transmit the data immediately.

VTAM flushes the buffer only when there is something in it. Issuing this macroinstruction when the SEND buffer is empty does not cause anything to flow to the partner LU.

For half-duplex conversations, VTAM buffers function management headers (FMH-5 and FMH-7). The FLUSH macroinstruction may be used to ensure that the headers are sent to the partner LU immediately.

Issuing an APPCCMD CONTROL=SEND, QUALIFY=FLUSH on a full-duplex conversation may cause the early completion of an APPCCMD CONTROL=RECEIVE, FILL=BUFF for the partner transaction program.

This macroinstruction corresponds to the FLUSH verb described in the LU 6.2 architecture.

**Context**
For half-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND
- PENDING_SEND

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD CONTROL=SEND, QUALIFY=FLUSH
```

 LU 6.2 macroinstruction syntax and operands 441
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA= rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB= acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.
CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
When the application program regains control after issuing an APPCCMD asynchronously, it is
prevented from issuing another APPCCMD against the same conversation resource (that
processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if
the conversation is full-duplex) until the command has completed. The exceptions to this are the
APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD
CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against
the same conversation resource that processes on the RECEIVE (if the conversation is full-
duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information
about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2
Guide

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY
is recommended when issuing the APPCCMD on a full-duplex conversation.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application
under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field
of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the
application under the same SRB in which VTAM was invoked. The indicator resides within the
RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the
processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled
RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'
SEND
X'02'
RECEIVE
X'03'
RECEIVE_CONFIRM
X'04'
RECEIVE_CONFIRM_SEND
X'05'
RECEIVE_CONFIRM_DEALLOCATE
X'07'
PENDING_END_CONVERSATION_LOG
X'08'
END_CONVERSATION
X'09'
PENDING_SEND
X'0A'
PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:
X'80'
  FDX_RESET
X'81'
  SEND/RECEIVE
X'82'
  SEND_ONLY
X'83'
  RECEIVE_ONLY
X'84'
  PENDING_SEND/RECEIVE_LOG
X'85'
  PENDING_RECEIVE-ONLY_LOG
X'86'
  PENDING_RESET_LOG

FDB2
  The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
  The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
  The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

  YES (B'1')
  One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

  NO (B'0')
  No FMH-5s are waiting to be received by the application program.

RCPRI
  The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
  The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
  The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
  Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.
State changes
No state changes are associated with this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_TIME</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>ERROR_INDICATION_RECEIVED—ALLOCATION_ERROR</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>ERROR_INDICATION_RECEIVED—UNKNOWN_ERROR_CODE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT_ REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
Purpose

This macroinstruction notifies the partner LU on a half-duplex conversation that the local application program is requesting to enter SEND state. The conversation is changed to SEND state when the local application program subsequently receives a SEND indication from the remote application program.

Usage

This macroinstruction can only be used on half-duplex conversations.

When this macroinstruction is issued, VTAM sends a SIGNAL RU to the partner application program to indicate that the local LU is requesting to enter SEND state. The signal code and signal extension fields of the SIGNAL RU carry X'00010001'.

When the partner application receives the REQUEST_TO_SEND notification, it can enter RECEIVE state. When an APPCCMD CONTROL=RECEIVE macroinstruction completes and the SEND indicator is on in the WHATRCV field, the local application program is informed that it is in SEND state.

This macroinstruction can be issued while other macroinstructions are outstanding. However, if this macroinstruction is issued while an APPCCMD CONTROL=SEND, QUALIFY=RQSEND or an APPCCMD CONTROL=SENDXPD macroinstruction is outstanding, it completes with return codes that indicate PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING.

This macroinstruction corresponds to the REQUEST_TO_SEND verb described in the LU 6.2 architecture.

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- SEND
- PENDING_SEND

This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
APPCCMD name  ——— CONTROL = ——— SEND ——— QUALIFY ——— =

          ——— RQSEND 1

          , ——— RPL = ——— rpl_address_field

          ( ——— rpl_address_register ——— )

          , ——— AAREA = ——— rpl_extension_address_field

          ( ——— rpl_extension_address_register ——— )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
EXIT is meaningful only for asynchronous operations.
You can code more than one suboperand on OPTCD, but no more than one from each group.
KEEPSRB is meaningful only for synchronous operations.
NKEEPSRB is meaningful only for synchronous operations.

Input parameters
The following information shows descriptions of the input parameters:

AAREA=\texttt{rpl\_extension\_address\_field}
AAREA=\texttt{(rpl\_extension\_address\_register)}
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\texttt{acb\_address\_field}
ACB=\texttt{(acb\_address\_register)}
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.
CONVID=32-bit resource_id_field
CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the EXPEDITED SEND queue until the command has completed. The application can issue APPCCMDs against the same conversation resource that processes on the SEND/RECEIVE if the conversation is half-duplex, or the SEND and RECEIVE queues if the conversation is full-duplex, and the EXPEDITED RECEIVE and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide
The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOOPT11 field of the RPL.

OPTCD=NEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOOPT11 field of the RPL.

RPL=\texttt{rpl\_addresses\_field} 
\texttt{RPL=\langle rpl\_address\_register \rangle}
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

\textbf{RPL and RPL extension fields modified by macroinstruction}

The following information shows descriptions of RPL and RPL extension fields:

\textbf{CONSTATE}

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

\begin{itemize}
  \item \texttt{X'01'} SEND
  \item \texttt{X'02'} RECEIVE
  \item \texttt{X'03'} RECEIVE\_CONFIRM
  \item \texttt{X'04'} RECEIVE\_CONFIRM\_SEND
  \item \texttt{X'05'} RECEIVE\_CONFIRM\_DEALLOCATE
  \item \texttt{X'06'} PENDING\_DEALLOCATE
  \item \texttt{X'07'} PENDING\_END\_CONVERSATION\_LOG
  \item \texttt{X'08'} END\_CONVERSATION
  \item \texttt{X'09'} PENDING\_SEND
  \item \texttt{X'0A'} PENDING\_RECEIVE\_LOG
\end{itemize}

For full-duplex conversations, this field can have the following values:

\begin{itemize}
  \item \texttt{X'80'} FDX\_RESET
  \item \texttt{X'81'} SEND\_RECEIVE
  \item \texttt{X'82'} SEND\_ONLY
\end{itemize}
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**
  No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**
Specifies 4 bytes of user data that the application requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
No state changes are associated with this macroinstruction.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED—CONTROL/QUALIFY_VALUE_INVALID_FOR_FULL-DUPLEX_CONVERSATION</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0005'</td>
<td>REQUEST_NOT_ALLOWED—RSP_HAS_NOT_BEEN_RECEIVED_FOR_A_PREVIOUS_SENDEXPDREQUEST</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR.OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SENDEXPD, QUALIFY=DATA**

**Purpose**

This macroinstruction sends expedited data to a partner LU over a full-duplex or a half-duplex conversation established on a full-duplex-capable session. If the session is not full-duplex capable, an RCPRI, RCSEC combination of X'00A0', X'0001', REQUEST_NOT_ALLOWED—LUPAIR_DOES_NOT_SUPPORT_SENDING_EXPEDITED_DATA is returned to the application.
Usage

The amount of expedited data specified by the application should be in the range of 1–86 bytes. If the length of the expedited data is outside of this range, an RCRPI, RCSEC combination of X'002C', X'002C', PARAMETER_ERROR—INVALID_EXPEDITED_DATA_LENGTH is returned to the application.

This macroinstruction will be posted complete immediately without waiting for a response from the partner LU. A response will not be sent by the partner until the expedited data has been received by the partner application.

If the conversation ends before the macroinstruction has a chance to process, RCRPI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application.

If this macroinstruction is issued while another APPCCMD CONTROL=SENDEXPD macroinstruction or an APPCCMD CONTROL=SEND, QUALIFY=RQSEND macroinstruction is currently outstanding for the specified conversation, an RCRPI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application.

If the EXPEDITED SEND queue has been prohibited, then an RCRPI, RCSEC combination of X'00A0', X'0002', REQUEST_NOT_ALLOWED—REQUEST_BLOCKED, is returned to the application.

An RCRPI, RCSEC combination of X'0050', X'0000', STATE_ERROR, will be returned when the macroinstruction is issued in PENDING_DEALLOCATE state.

If the macroinstruction is issued and the response to a previously issued SENDEXPD request has not been received, then an RCRPI, RCSEC combination of X'00A0', X'0005', REQUEST_NOT_ALLOWED—RSP_HAS_NOT_BEEN_RECEIVED_FOR_A_PREVIOUS_SENDEXPD_REQUEST is returned to the application.

This macroinstruction will always cause a flow.

This macroinstruction corresponds to the SEND_EXPEDITED_DATA verb described in the LU 6.2 architecture.

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PENDING_END_CONV_LOG
- PENDING_SEND
- PENDING_RECEIVE_LOG

For full-duplex conversation, this macroinstruction can be issued from the following conversations states:

- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.
Syntax

APPCCMD — — CONTROL — — SENDEXPD — — QUALIFY — —

DATA

RPL — — rpl_address_field

AAREA — — rpl_extension_address_field

ACB — — acb_address_field

AREA — — data_area_or_buffer_list_address_field

BRANCH — — NO

CONMODE — — BUFFCA

CONVID — — 32-bit_resource_id_field

CONXMOD — — CA
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=data_area_or_buffer_list_address_field
AREA=(data_area_or_buffer_list_address_register)

Specifies the address of a data buffer or buffer list. If OPTCD=NBUFFLST, AREA specifies the address of a data area containing the data to be sent. If OPTCD=BUFFLST, AREA specifies the address of a buffer list that in turn points to the data to be sent. This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME

Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address-field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address-field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLEXTDS field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the EXPEDITED SEND queue until the command has completed. The application can issue APPCCMDs against the same conversation resource that processes on the SEND/RECEIVE if the conversation is half-duplex, or the SEND and RECEIVE queues if the conversation is full-duplex, and the EXPEDITED RECEIVE and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

RECLEN=data_length
RECLEN=(data_length_register)
Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

• If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the buffer area specified by AREA.
• If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. The RECLEN specifies a buffer list length that is a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'
SEND
X'02'
RECEIVE
X'03'
RECEIVE_CONFIRM
X'04'
RECEIVE_CONFIRM_SEND
X'05'
RECEIVE_CONFIRM_DEALLOCATE
X'06'
PENDING_DEALLOCATE
X'07'
PENDING_END_CONVERSATION_LOG
For full-duplex conversations, this field can have the following values:

- **X'80'**
  - FDX_RESET
- **X'81'**
  - SEND/RECEIVE
- **X'82'**
  - SEND_ONLY
- **X'83'**
  - RECEIVE_ONLY
- **X'84'**
  - PENDING_SEND/RECEIVE_LOG
- **X'85'**
  - PENDING_RECEIVE-ONLY_LOG
- **X'86'**
  - PENDING_RESET_LOG

**EXPDLLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  - One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**
  - No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.
RCSEC  
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD  
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SIGDATA  
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

SIGRCV  
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B'1')  
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')  
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD  
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.

State changes  
No state changes are associated with this macroinstruction.

Return codes  
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=SENDFMH5, QUALIFY=NULL**

**Purpose**

This macroinstruction accepts and sends an FMH-5 for a conversation reserved by the APPCCMD CONTROL=PREALLOC macroinstruction.

**Usage**

This macroinstruction completes the allocation of a conversation begun by a previous APPCCMD CONTROL=PREALLOC. VTAM does not activate any additional session between the application program and its partner LU as a result of this command.

The APPCCMD CONTROL=SENDFMH5 macroinstruction does not return any vectors to the application in the vector area. For conversations on half-duplex-capable sessions, the FMH-5 is stored in the SEND buffer. For conversations on full-duplex-capable sessions, the FMH-5 is flushed immediately.

**Context**

This macroinstruction can only be issued from the PENDING_ALLOCATE conversation state.

**Syntax**

```
APPCCMD -- -- CONTROL -- = -- SENDFMH5 -- , -- QUALIFY -- =
```

```
name
```

```
NULL 1
```
RPL = rpl_address_field
   ( rpl_address_register )

AAREA = rpl_extension_address_field
   ( rpl_extension_address_register )

ACB = acb_address_field
   ( acb_address_register )

AREA = fmh-5_and_optional_pip_gds_variable_address_field
   ( fmh-5_and_optional_pip_gds_variable_address_register )

BRANCH = NO
       YES

CONVID = 32-bit_resource_id_field
   ( 32-bit_resource_id_register )

ECB = INTERNAL
   ecb_address_field
   ( ecb_address_register )

EXIT = exit_routine_address_field
   ( exit_routine_address_register )

OPTCD = ( ASY
        SYN
        KEEPSRB
        NKEEPSRB

RECLEN = fmh-5_and_optional_pip_gds_variable_length
   ( fmh-5_and_optional_pip_gds_variable_length_register )

Notes:

1Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

ECB is meaningful only for asynchronous operations.

EXIT is meaningful only for asynchronous operations.

You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPSRB is meaningful only for synchronous operations.

**Input parameters**

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=fmh-5_and_optional_pip_gds_variable_address_field**

AREA=(fmh-5_and_optional_pip_gds_variable_address_register)

Specifies the address of a data buffer or buffer list. If OPTCD=NBUFLST, AREA specifies the address of a data area containing the data to be sent. If OPTCD=BUFLST, AREA specifies the address of a buffer list that in turn points to the data to be sent. In either case, the data consists of logical records. VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.) This field is labeled RPLAREA in the RPL.

**BRANCH**

 Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**
  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**

  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID=32-bit_resource_id_field**

CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

- **ECB=INTERNAL**

  Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.
**ECB=ecb_address_field**

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=fmh-5_and_optional_gds_field_length**

RECLEN=(fmh-5_and_optional_gds_field_length_register)

Specifies the length of the data within the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**

RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

**CONSTATE**

The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.

This field can have the following values for half-duplex conversations:

**X'00'**

RESET

**X'01'**

SEND

**X'08'**

END_CONVERSATION

This field can have the following values for full-duplex conversations:
X'00'  
 RESET
X'80'  
FDX_RESET
X'81'  
SEND/RECEIVE

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.
State changes
After successful completion of this macroinstruction, the conversation state is SEND if issued over a half-
duplex session or SEND/RECEIVE if issued over a full-duplex session.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues
this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these
return codes.

<table>
<thead>
<tr>
<th>RC PRI</th>
<th>RC SEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0006’</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0000’</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0010’</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_VALID_FOR_FULL-DUPLEX</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0011’</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FDX_CONVERSATION</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0002’</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000A’</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMH5_SUPPLIED</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000F’</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0010’</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0011’</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0015’</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’001F’</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0022’</td>
<td>PARAMETER_ERROR—INVALID_CONTROL_OR_QUALIFY_VALUE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0032’</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X’0048’</td>
<td>X’0000’</td>
<td>RESOURCE_FAILURE,_NO_RETRY</td>
</tr>
<tr>
<td>X’004C’</td>
<td>X’0000’</td>
<td>RESOURCE_FAILURE,_RETRY</td>
</tr>
<tr>
<td>X’0050’</td>
<td>X’0000’</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X’0070’</td>
<td>X’0000’</td>
<td>TEMPORARY_STORAGE_SHORTAGE OR RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X’0074’</td>
<td>X’0000’</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X’0078’</td>
<td>X’0000’</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X’007C’</td>
<td>X’0000’</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X’0090’</td>
<td>X’0000’</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=SENDRCV, QUALIFY=DATAFLU

Purpose
This macroinstruction provides a dual function; it performs the function of a send, and when the send is
complete it automatically performs the function of a receive.

The send portion of this macroinstruction sends data supplied by the application program and any data
that is already in the SEND buffer to the partner application. After the send portion of this
When this macroinstruction is issued, VTAM places data in the SEND buffer of the conversation that is specified by the CONVID parameter. VTAM determines the location of the data to be sent from the buffer list entries specified by the AREA parameter. VTAM sends all data in the SEND buffer to the partner LU.

When the send portion of this macroinstruction completes, there is no data ready to be received on the conversation; therefore, VTAM queues the macroinstruction until data arrives. This macroinstruction has just turned the flow around and the SEND indication is still enroute to the partner. After the partner receives the data just sent and also the SEND indication, it may then send data back to the local application. When enough of this data is received by VTAM to satisfy the receive portion of this macroinstruction the macroinstruction will be completed.

After data is received, VTAM copies any received data from the conversation that is specified by the CONVID parameter to the data area that is specified by the last entry in the buffer list.

When this macroinstruction completes, the BLERECLN field of the last buffer list entry indicates how much data was written to the data area. The WHATRCV field indicates what type of data was received.

The application program can issue this macroinstruction when the conversation is in SEND or PENDING_SEND state. VTAM flushes its SEND buffer, sending all buffered information, along with the SEND indicator, to the partner LU. This changes the conversation to RECEIVE state. VTAM then waits for information to arrive. The remote application program can send data to the local application program after it receives the SEND indication.

For a complete discussion of sending data and receiving data, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

Context
For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

• SEND
• PENDING_SEND

For full-duplex conversations, this macroinstruction is not allowed.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
name APPCCMD — — CONTROL — = — SENDRCV — , — QUALIFY — = —
DATAFLU 1

, — RPL — = — rpl_address_field — ( — rpl_address_register — )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. Refer to "Coding Default Values" in z/OS Communications Server: SNA Programmer's LU 6.2 Guide for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=rpl_extension_address_register
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=acb_address_register
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=buffer_list_address_field
AREA=buffer_list_address_register
Specifies the address of a list of buffer entries.

• If OPTCD=BUFFLST, all entries in the buffer list except the last specify the address and length of data to be sent. The data consists of logical records. VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

The last entry specifies the address and length of an area in which data is to be received. When this macroinstruction completes, another field in this last entry contains the number of bytes placed in this receive buffer by VTAM.
Both the send and receive buffers are described using the ISTBLEN T DSECT. For a more detailed description of how to use buffer list entries refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

- If OPTCD=XBUFLST, all entries in the buffer list except the last specify the address and length of data to be sent. The send data resides in CSM buffers. VTAM does not track logical records supplied by the application.

Like OPTCD=BUFLST, the last entry specifies the address and length of an area in which data is to be received. When this macroinstruction completes, another field in this last entry contains the number of bytes placed in this receive buffer by VTAM. This receive buffer is not a CSM buffer.

The send buffers are described using the ISTBLXEN DSECT and the receive buffer is described using the ISTBLEN T DSECT. For a more detailed description of how to use extended buffer list entries, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH= YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CD
This parameter controls subsequent actions if a SEND indication is received in the WHATRCV field on the receive portion of this macroinstruction. For this to happen, the send portion of this macroinstruction transmitted the SEND indication to the partner, as is normally done on this macroinstruction, which in turn returned it. The SEND indication is being reported back to the local application on the receive portion of this macroinstruction. In particular CD specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.

CD=DEFER
Specifies that the conversation state will be in PEND_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CD=IMMED
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this
conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=32-bit_resource_id_register**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=exit_routine_address_register**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**FILL**

Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter applies only to the receive portion of this macroinstruction and corresponds to FILL=LL|BUFFER described in the LU 6.2 architecture. This field is labeled RPL6FILL in the RPL extension.

**FILL=BUFF**

Specifies the application program is to receive data independently of its logical-record format. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.
**FILL=LL**

Specifies the application program is to receive one logical record, or whatever portion of the logical record is available. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

In general, when the application program regains control after issuing an asynchronous APPCCMD, it is prevented from issuing another APPCCMD against the same conversation resource until the prior asynchronous command has completed. The exceptions to this are the APPCCMD CONTROL=SEND, QUALIFY=RQSEND; APPCCMD CONTROL=REJECT; and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. (For more information, refer to the descriptions of the particular macroinstructions). The application program is allowed to issue APPCCMDs against other conversations.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=BUFFLST**

Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries. This field is labeled RPLBUFFL in the RPL. When OPTCD=BUFFLST, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The buffer list created by the application must have at least two entries. One or more entries must be send buffer list entries. This specifies the layout of the send buffers. The last entry must be a special receive entry that points to the receive buffer and indicates the area length. Both the send buffer(s) and the receive buffer are described by the ISTITBLENT macroinstruction. The following list explains the layout of the receive entry:

- The first 4 bytes are reserved and should be set to 0 when the macroinstruction is issued. This field will be used to return the amount of data received to the application.
- The second 4 bytes contain the length of the receive buffer. This is similar to the AREALEN field of an RPL that accompanies a receive type macroinstruction.
- The third 4 bytes contain the address of a receive buffer. This is similar to the AREA field that accompanies a receive type macroinstruction.
- The fourth 4 bytes must contain zeros (the send length field).
**OPTCD=XBUFLST**

Specifies that the data supplied by the application program is contained within an extended buffer list. The AREA field of the RPL points to an extended buffer list that contains a contiguous set of 48-byte send extended buffer list entries followed immediately by a 16-byte receive buffer entry. Once OPTCD=XBUFLST has been issued, VTAM no longer tracks logical records for the duration of the conversation.

The indicator is labeled RPLXBFBL and resides within the RPLOPT6 field of the RPL.

Each send entry in the extended buffer list can point to any displacement into a CSM buffer and is described by ISTBLXEN. VTAM uses the CSM token rather than the storage address to track a CSM buffer. A CSM token cannot be repeated in an extended buffer list. If multiple areas of a CSM buffer are to be used on one APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction. This macroinstruction returns a new token for each CSM buffer segment. The new tokens should then be used on the APPCCMD. VTAM treats the CSM storage associated with the new CSM tokens as separate CSM buffers.

The last entry describes the receive buffer. This buffer is not a CSM buffer. It is described using the ISTBLENT DSECT.

**RECLLEN**

Specifies the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=BUFFLST, the length of the buffer list is determined by the product of 16 and the number of entries, both send and receive. (Each buffer list entry consists of 16 bytes.)
- If OPTCD=XBUFLST, the length of the buffer list is determined by the product of 48 and the number of send entries plus 16 bytes for the receive buffer entry. (Each CSM buffer list entry consists of 48 bytes.)

**RPL=rpl_address_field**

**RPL=rpl_address_register**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

The following conversation states are possible:

- X’01’ SEND
- X’02’ RECEIVE
- X’03’ RECEIVE CONFIRM
- X’04’ RECEIVE CONFIRM_SEND
- X’05’ RECEIVE CONFIRM_DEALLOC
- X’07’ PENDING_END_CONVERSATION_LOG
- X’08’ END CONVERSATION
**EXPDLLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**
  No FMH-5s are waiting to be received by the application program.

**LOGRCV**
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

- **YES (B'1')**
  An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- **X'0004'**
  ALLOCATION_ERROR
- **X'0014'**
  DEALLOCATE_ABEND_PROGRAM
- **X'0018'**
  DEALLOCATE_ABEND_SERVICE
- **X'001C'**
  DEALLOCATE_ABEND_TIMER
- **X'0030'**
  PROGRAM_ERROR_NO_TRUNC
X'0034'
  PROGRAM_ERROR_PURGING
X'0038'
  PROGRAM_ERROR_TRUNC
X'003C'
  SERVICE_ERROR_NO_TRUNC
X'0040'
  SERVICE_ERROR_PURGING
X'0044'
  SERVICE_ERROR_TRUNC
X'0048'
  RESOURCE_FAILURE_NO_RETRY
X'005C'
  USER_ERROR_CODE_RECEIVED

NO (B'0')
 Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
 The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
 The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
 The field used on the SEND portion of this macroinstruction, similar to a send with a buffer list. It is used to calculate the number of entries in the buffer list.

For the receive portion of this macroinstruction, VTAM calculates a RECLEN value but does not overlay the RECLEN provided by the application in the RPL. Instead, VTAM returns the receive RECLEN in the first 4 bytes of the last entry in the buffer list (BFERECLN), which is the entry used to describe the receive area.

RPLXSRV
 A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
 The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
 The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates
USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. It is labeled RPL6NSNI in the RPL extension.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG.

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**STSHBF**
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

**WHATRCV**
The field in the RPL extension that returns a mask specifying what the application program received. It is labeled RPL6WHAT. The application program should examine this WHATRCV mask only when RCPRI indicates X’0000’. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B’1’) to indicate the type of information that has been received. For instance, if the application program is
being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format.

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>RPL6RCV2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Meaning</td>
</tr>
<tr>
<td>0</td>
<td>DATA</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X’8000’. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a discussion of the meaning of this field.

**State changes**

See the description of the WHATRCV mask for state changes when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 533 for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000'</td>
<td>X’0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0006'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0007'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0008'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X’0004'</td>
<td>X’000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X’0014'</td>
<td>X’0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X’0018'</td>
<td>X’0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0013'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0014'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0016'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0017'</td>
<td>PARAMETER_ERROR—SENDRCV_SPECIFIED_WITHOUT_OPTCD=BUFFLST</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0018'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=SETSESS, QUALIFY=RESUME

Purpose
This macroinstruction resumes sending any outgoing normal data that was held because APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND was issued previously on the specified session.

Usage
This macroinstruction should be issued to notify VTAM to allow any outbound normal data to flow to the partner if any has been held due to a previously issued APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND command. APPCCMD CONTROL=SETSESS, QUALIFY=RESUME also enables the following items to resume:

• Normal data flow from any conversations matched to the session
• Normal session deactivation
• Session bidding

This macroinstruction indicates to VTAM that the application program (which is supporting a sync point manager) has completed its synchronization processing successfully.

APPCCMD CONTROL=REJECT, QUALIFY=SESSION can be issued if the application program's synchronization processing was unsuccessful and the application program does not wish to imply by the normal data flow that the sync point completed successfully.

If this macroinstruction is issued and the session has not been suspended, a return code of 0 is received, but no changes are made.

Context
This macroinstruction is not conversation-specific and therefore is not conversation-state-driven.

Syntax

```
APPCCMD  \\
  CONTROL = SETSESS  \\
  QUALIFY = RESUME
```

**APPCCMD CONTROL=SETSESS, QUALIFY=RESUME**

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>CONTROL/QUALIFY_VALUE_INVALID_FOR_FULL-DUPLEX_CONVERSATION</td>
<td>X'00A0' X'0004'</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>REQUEST_NOT_ALLOWED—PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
<td>X'00A0' X'0006'</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT REQUESTED_FUNCTION</td>
<td>X'00A8' X'0000'</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
<td>X'00A8' X'0001'</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
<td>X'00A8' X'0002'</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
<td>X'00B4' X'0001'</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0002'</td>
<td>CSM_DETECTED_ERROR—INVALID_BUFFER_TOKEN_SPECIFIED</td>
<td>X'00B4' X'0002'</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0003'</td>
<td>CSM_DETECTED_ERROR—INVALID_INSTANCE_ID_SPECIFIED</td>
<td>X'00B4' X'0003'</td>
</tr>
</tbody>
</table>
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
EXIT is meaningful only for asynchronous operations.

You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
Specifies the session to which this macroinstruction applies. The session instance identifier, which was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction, indicates the session to be released. This field is labeled RPL6SSID in the RPL extension.

SESSIDL=session_instance_id_length
SESSIDL=(session_instance_id_length_register)
Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
Following are descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

State changes
No state changes are associated with this macroinstruction.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>PARAMETER_ERROR—INVALID_SESSION_INSTANCE_IDENTIFIER</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND

Purpose

This macroinstruction specifies that the application program wants VTAM to suspend any outgoing normal data flow on the specified session after the current conversation has been deallocated. APPCCMD CONTROL=SETSESS, QUALIFY=RESUME resumes the outgoing normal flow.

Usage

This macroinstruction should be issued to notify VTAM to not allow outbound flow on the session. It should be issued if the application program (which is supporting a sync point manager) has not completed the synchronization processing needed before the partner can continue its synchronization processing. The application program must issue this command before the conversation supporting the sync point exchange is deallocated to ensure the flow is stopped on the free session.

Suspending the session gives the application program with the sync point manager control of the outbound flow whose subsequent receipt at the partner implies a successful sync point has completed. The partner application program can then continue synchronization cleanup. Further information on the sync point services function is described in the SNA Format and Protocol Reference Manual: Architecture Logic for LU Type 6.2.

APPCCMD CONTROL=SETSESS, QUALIFY=RESUME indicates that the application program is ready to resume normal flow because its sync point processing completed successfully. APPCCMD CONTROL=REJECT, QUALIFY=SESSION can be issued if the sync point processing is unsuccessful.

If an application program is executing under persistent LU-LU session support and the application program fails after APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND has been issued and APPCCMD CONTROL=SETSESS, QUALIFY=RESUME has not been issued, VTAM UNBINDs the session and
deallocates the conversation on which the synchronization is taking place. In the same situation, VTAM also UNBINDs sync point sessions for which APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEBEG has been issued but neither APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND nor APPCCMD CONTROL=SETSESS, QUALIFY=RESUME has been issued at the time of the failure.

**Context**

This macroinstruction is not conversation-specific and, therefore, is not driven by the conversation state. This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
name
```

```appcmd  control  =  setsess
qualify  =  suspend
```

```
,  rpl  =  rpl_address_field
     (  rpl_address_register  )
```

```
,  aarea  =  rpl_extension_address_field
     (  rpl_extension_address_register  )
```

```
,  acb  =  acb_address_field
     (  acb_address_register  )
```

```
,  branch  =  no
    yes
```

```
,  convid  =  32-bit_resource_id_field
     (  32-bit_resource_id_register  )
```

```
,  ecb  =  internal
```

```
,  exit  =  exit_routine_address_field
     (  exit_routine_address_register  )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit Routine_address_field
EXIT=(exit_Routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
Specifies the session to which this macroinstruction applies. The session instance identifier, which was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD
CONTROL=RCVFMH5 macroinstruction, indicates the session to be held. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL=session_instance_id_length**

**SESSIDL=(session_instance_id_length_register)**

Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. This field is labeled RPL6SSID in the RPL extension.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

### State changes

There are no state changes associated with this macroinstruction.

### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, "Return codes," on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>PARAMETER_ERROR—INVALID_SESSION_INSTANCE_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0026'</td>
<td>PARAMETER_ERROR—SESSION_INSTANCE_IDENTIFIER_AND_CONVERSATION_ID_ARE_MISMATCHED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=SETSESS, QUALIFY=SYNCBEG

#### Purpose

This macroinstruction notifies VTAM that a sync point exchange is beginning.

If an application program is executing under persistent LU-LU session support, persistence must be overridden for a session during the time that a sync point exchange takes place. If the application program fails after APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND has been issued and APPCCMD CONTROL=SETSESS, QUALIFY=RESUME has not been issued, VTAM UNBINDs the session and deallocates the conversation on which the synchronization is taking place. In the same situation, VTAM also UNBINDs sync point sessions for which APPCCMD CONTROL=SETSESS, QUALIFY=SYNCBEG has been issued, but neither APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND nor APPCCMD CONTROL=SETSESS, QUALIFY=RESUME has been issued at the time of the failure.

#### Usage

This macroinstruction is issued to notify VTAM that the sync point manager is beginning a synchronization exchange because a SYNCPT is being issued or a TAKE-SYNCPT is being received. To ensure that synchronization protocols are followed, VTAM UNBINDs this session when the application program fails, even though the application program has enabled persistence. The UNBIND permits the LUs to make consistent decisions and ensures continued synchronization between the two LUs. If the data is critical enough to use a synchronization exchange, APPCCMD CONTROL=SETSESS, QUALIFY=SYNCBEG and APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND should be used. For circumstances for use, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

#### Context

This macroinstruction is not conversation-specific and, therefore, is not driven by the conversation state. It performs a useful function only for application programs that are using persistent LU-LU sessions. If application programs that have not enabled persistence issue this macroinstruction, a good return code is sent but no action is taken.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

#### Syntax

```
APPCCMD CONTROL=SETSESS, QUALIFY=SYNCBEG
```

### Table: RCPRI and RCSEC Codes

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=

**AAREA=**(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=

**ACB=**(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID**=

**CONVID=**(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.
EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
Specifies the session to which this macroinstruction applies. The session instance identifier, which was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFHM5 macroinstruction, indicates the session to be released. This field is labeled RPL6SSID in the RPL extension.

SESSIDL=session_instance_id_length
SESSIDL=(session_instance_id_length_register)
Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFHM5 macroinstruction. This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is

LU 6.2 macroinstruction syntax and operands 493
labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**State changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>INVALID_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>PARAMETER_ERROR—INVALID_SESSION_INSTANCE_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0026'</td>
<td>SESSION_INSTANCE_IDENTIFIER_AND_CONVERSATION_IDEN_MISMATCH</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

---

**APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND**

**Purpose**

This macroinstruction indicates to VTAM that the sync point exchange has completed.

**Note:** This macroinstruction only has meaning for MVS and VSE applications using persistent sessions. VTAM ignores this macroinstruction if issued from a VM application.
Usage
This macroinstruction is issued to notify VTAM that the sync point exchange has completed, whether successful or not, and that VTAM no longer needs to UNBIND sync point sessions during a failure after persistence has been enabled. It is used with APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND. For circumstances for use, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

Context
This macroinstruction is not conversation-specific and, therefore, is not driven by conversation state. It performs a useful function only for application programs that are using persistent LU-LU sessions. If application programs that have not enabled persistence issue this macroinstruction, a good return code is sent but no action is taken.

Syntax

```
APPCCMD  CONTROL = SETSESS , QUALIFY = SYNCEND

, RPL = rpl_address_field
  ( rpl_address_register )

, AAREA = rpl_extension_address_field
    ( rpl_extension_address_register )

, ACB = acb_address_field
  ( acb_address_register )

, BRANCH = NO

, ECB = INTERNAL
  ( ecb_address_field )

, EXIT = exit_routine_address_field
  ( exit_routine_address_register )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** *rpl_extension_address_field*

AAREA=*(rpl_extension_address_register)*

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** *acb_address_field*

ACB=*(acb_address_register)*

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
Specifies the session to which this macroinstruction applies. The session instance identifier, which was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction, indicates the session to be released. This field is labeled RPL6SSID in the RPL extension.
SESSIDL = \text{session\_instance\_id\_length}
SESSIDL = (\text{session\_instance\_id\_length\_register})

Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

- **FDB2**
  The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

- **RCPRI**
  The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

- **RCSEC**
  The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

- **RTNCD**
  The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

State changes

No state changes are associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO ECB FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL BLOCK INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>PARAMETER_ERROR—INVALID_SESSION_INSTANCE_IDENTIFIER</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL

**Purpose**

This macroinstruction obtains status on information from any active conversation. VTAM will wait for information to arrive on a conversation to satisfy the macroinstruction request. If information is available to be received, the application will receive status on the information without waiting.

**Usage**

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be specified in the RPLAREA field which can be set with the AREA keyword. See “Status data structure (ISTSTATD)” on page 589 for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be a least 48 bytes) an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.

Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

- Normal information
- Expedited information (data and/or Request_To_Send Received)

If this macroinstruction is issued while another TESTSTAT ALL|IALL is currently outstanding, an RCPRI, RCSEC combination of X'0002C', X'0008', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application program.

This macroinstruction will not alter the conversation.

**Context**

Input states are not applicable to this macroinstruction.

**Syntax**

```
name APPCCMD — CONTROL = TESTSTAT , QUALIFY = ALL

RPL = rpl_address_field

AAREA = rpl_extension_address_field
```

---

**RCPRI**  |  **RCSEC**  |  **Meaning**
---|---|---
X'00A8'  |  X'0001'  |  ENVIRONMENT_ERROR—SUSPEND_FAILURE
X'00A8'  |  X'0002'  |  ENVIRONMENT_ERROR—RESUME_FAILURE
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters
The following information shows descriptions of the input parameters:
AAREA=\textit{rpl\_extension\_address\_field}
AAREA=(\textit{rpl\_extension\_address\_register})

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\textit{acb\_address\_field}
ACB=(\textit{acb\_address\_register})

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=\textit{data\_area\_address\_field}
AREA=(\textit{data\_area\_address\_register})

Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, CITY-STATE. This field is labeled RPLAREA in the RPL.

AREALEN=\textit{data\_area\_length}
AREALEN=(\textit{data\_area\_length\_register})

Specifies the length value that is the maximum amount of data the application program is to receive. The application program must receive at least 48 bytes of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=\textit{internal}\n
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=\textit{ecb\_address\_field}\nECB=(\textit{ecb\_address\_register})

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. \textit{Event\_control\_block\_address} is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=\textit{exit\_routine\_address\_field}\nEXIT=(\textit{exit\_routine\_address\_register})

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the
function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the
RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the
application program is to be informed later of the completion of the macroinstruction by the
posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of
the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application
under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field
of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the
application under the same SRB in which VTAM was invoked. The indicator resides within the
RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the
processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application
program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the
application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of
the longest FMH-5 to be received by the application program. This field has meaning only when
FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The
indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL
extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set
to YES so long as an FMH-5 is waiting to be received by the application program. The application
program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the
application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is
labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the
application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is
labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the
result of the macroinstruction processing.
**RECLLEN**

The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI,RCSEC fields equal X'0000', X'0000'. If the RCPRI,RCSEC fields equal X'002C', X'0008' RECLLEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLEN in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRRTNCD.

**State Changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'007B'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=TESTSTAT, Qualifiable**

**Purpose**

This macroinstruction obtains status on information immediately available from any active conversation. VTAM will not wait for information to arrive on a conversation to satisfy the macroinstruction request.
Usage

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be specified in the RPLAREA field, which can be set with the AREA keyword. See “Status data structure (ISTSTATD)” on page 589 for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be at least 48 bytes), an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.

If this macroinstruction is issued and information is not available on any conversation, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application.

Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

- Normal information
- Expedited information (data and/or Request_To_Send Received)

If this macroinstruction is issued while another TESTSTAT ALL|IALL is currently outstanding, an RCPRI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application program.

This macroinstruction will not alter the conversation.

Context

Input states are not applicable to this macroinstruction.

Syntax

```
APPCMD  CONTROL  =  TESTSTAT  ,  QUALIFY  =  
 IALL 1

,  RPL  =  rpl_address_field
 (  rpl_address_register  )

,  AAREA  =  rpl_extension_address_field
 (  rpl_extension_address_register  )

,  ACB  =  acb_address_field
 (  acb_address_register  )

,  AREA  =  data_area_address_field
 (  data_area_address_register  )
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with
transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=data_area_address_field**

**AREA=(data_area_address_register)**

Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, CITY-STATE. This field is labeled RPLAREA in the RPL.

**AREALEN=data_area_length**

**AREALEN=(data_area_length_register)**

Specifies the length value that is the maximum amount of data the application program is to receive. The application program must receive at least 48 bytes of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLEXTDS field of the RPL.
**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B’1’)**
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B’0’)**
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI,RCSEC fields equal X’0000’, X’0000’. If the RCPRI,RCSEC fields equal X’0002’, X’0008’ RECLEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD.
**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=TESTSTAT, QUALIFY=ISPEC**

**Purpose**

This macroinstruction obtains status on information immediately available on a specified conversation. VTAM will not wait for information to arrive to satisfy the macroinstruction request.

**Usage**

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be specified in the RPLAREA field which can be set with the AREA keyword. See “Status data structure (ISTSTATD)” on page 589 for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be a least 48 bytes) an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.

If information is not available, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application program.

If the conversation ends before this macroinstruction can query the information received, if any, an RCPRI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application.
Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

- Normal information
- Expedited information (data and/or Request_To_Send Received)

This macroinstruction will not alter the conversation.

**Context**

This macroinstruction can be issued in any conversation state while the conversation is active so long as another APPCCMD CONTROL=TESTSTAT, QUALIFY=SPEC|ISPEC macroinstruction is not currently outstanding for the specified conversation.

**Syntax**

```
APPCCMD  name  CONTROL      =  TESTSTAT ,  QUALIFY     =  ISPEC
,  RPL  =  rpl_address_field
,  AAREA  =  rpl_extension_address_field
,  ACB  =  acb_address_field
,  AREA  =  data_area_address_field
,  AREALEN  =  data_area_length
,  BRANCH  =  NO  YES
,  CONVID  =  32-bit_resource_id_field
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=rpl_extension_address_field**

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=data_area_address_field**

**AREA=(data_area_address_register)**

Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, CITY-STATE. This field is labeled RPLAREA in the RPL.


**AREALEN**=
*
**AREALEN**=(data_area_length_register)

Specifies the length value that is the maximum amount of data the application program is to receive. The application program must receive at least 48 bytes of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the TESTSTAT queue until the command has completed. The application can issue APPCCMDs against the same conversation resource that processes on the SEND/RECEIVE queue if the conversation is half-duplex, or the SEND and RECEIVE queues if the conversation is full-duplex,
and the EXPEDITED RECEIVE and EXPEDITED SEND queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. For half-duplex conversations, this field can have the following values:

X’01’
SEND
X’02’
RECEIVE
X’03’
RECEIVE_CONFIRM
X’04’
RECEIVE_CONFIRM_SEND
X’05’
RECEIVE_CONFIRM_DEALLOCATE
X’06’
PENDING_DEALLOCATE
X’07’
PENDING_END_CONVERSATION_LOG
X’08’
END_CONVERSATION
X’09’
PENDING_SEND
X’0A’
PENDING_RECEIVE_LOG

For full-duplex conversations, this field can contain the following values:

X’80’
FDX_RESET
X’81’
SEND/RECEIVE
X’82’
SENDONLY
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B’1’)**
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B’0’)**
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI,RCSEC fields equal X’0000’, X’0000’. If the RCPRI,RCSEC fields equal X’002C’, X’0008’ RECLEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=TESTSTAT, QUALIFY=SPEC

Purpose

This macroinstruction obtains status on information available on a specified conversation. VTAM will wait for information to arrive to satisfy the macroinstruction request. If information is already available, the application program receives status on it without waiting.

Usage

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be specified in the RPLAREA field, which can be set with the AREA keyword. See “Status data structure (ISTSTATD)” on page 589 for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be at least 48 bytes), an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.
Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

- Normal information
- Expedited information (data and/or Request_To_Send Received)

If the conversation is terminated before information is received, an RCPRI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application program.

This macroinstruction will not alter the conversation.

**Context**

This macroinstruction can be issued in any conversation state while the conversation is active so long as another APPCCMD CONTROL=TESTSTAT QUALIFY=SPEC|ISPEC macroinstruction is not currently outstanding for the specified conversation.

**Syntax**

```plaintext
APPCCMD  --  CONTROL = TESTSTAT ,  QUALIFY = SPEC

,  RPL = rpl_address_field
     ( rpl_address_register )

,  AAREA = rpl_extension_address_field
     ( rpl_extension_address_register )

,  ACB = acb_address_field
     ( acb_address_register )

,  AREA = data_area_address_field
     ( data_area_address_register )

,  AREALEN = data_area_length
     ( data_area_length_register )

,  BRANCH = NO

,  CONVID = 32-bit_resource_id_field
     ( 32-bit_resource_id_register )
```
ECB = INTERNAL
( ecb_address_field )

EXIT = exit_routine_address_field
( exit_routine_address_register )

OPTCD = ( ASY 3 SYN 3 KEEPSRB 3 NKEEPSRB 3 7 8 )

Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters
The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
   Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
   Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=data_area_address_field
AREA=(data_area_address_register)
   Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, ISTSTATD. This field is labeled RPLAREA in the RPL.
AREALEN=data_area_length
AREALEN=(data_area_length_register)

Specifies the length value that is the maximum amount of data the application program is to receive. The application program must receive at least 48 bytes of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the TESTSTAT queue until the command has completed. The application can issue APPCCMDs against the same conversation resource that processes on the SEND/RECEIVE queue if the conversation is half-duplex, or the SEND and RECEIVE queues if the conversation is full-duplex,
and the EXPEDITED RECEIVE and EXPEDITED SEND queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'
SEND
X'02'
RECEIVE
X'03'
RECEIVE_CONFIRM
X'04'
RECEIVE_CONFIRM_SEND
X'05'
RECEIVE_CONFIRM_DEALLOCATE
X'06'
PENDING_DEALLOCATE
X'07'
PENDING_END_CONVERSATION_LOG
X'08'
END_CONVERSATION
X'09'
PENDING_SEND
X'0A'
PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

X'80'
FDX_RESET
X'81'
SEND/RECEIVE
X'82'
SEND_ONLY

X'83'
RECEIVE_ONLY

X'84'
PENDING_SEND/RECEIVE_LOG

X'85'
PENDING_RECEIVE-ONLY_LOG

X'86'
PENDING_RESET_LOG

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI, RCSEC fields equal X'0000', X'0000'. If the RCPRI, RCSEC fields equal X'002C', X'0008' RECLEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLLEN in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.
**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 533 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**ISTGAPPC**

**Purpose**

This macroinstruction declares and sets a list of global variables to indicate which LU 6.2 options are supported by the installed release of VTAM.

**Usage**

ISTGAPPC can be invoked directly, or by either IFGRPL or IFGACB as an inner macroinstruction call. The global variables defined for ISTGAPPC are shown in Table 2 on page 521.

To use the ISTGAPPC macroinstruction, the programmer must be familiar with the GBLA and SETA assembler language instructions, which are described in the assembler language publication for your operating system.

The use of ISTGAPPC is similar to the use of the ISTGLOBAL macroinstruction. For details, refer to the description of ISTGLOBAL in z/OS Communications Server: SNA Programmer’s LU 6.2 Reference.
The variables defined by ISTGAPPC are available to the application program at assembly time. If you want the application program to check these values at execution time, you can use the function-list vector described in the z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

Each global variable is an arithmetic symbol that can be set to 0, 1, or 2. The following information shows the meanings for the global variables and the corresponding levels of support.

**Global Variable**

<table>
<thead>
<tr>
<th>Support Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
</tr>
<tr>
<td>No (Option is not supported.)</td>
</tr>
<tr>
<td>X'01'</td>
</tr>
<tr>
<td>Yes (Option is supported.)</td>
</tr>
<tr>
<td>X'02'</td>
</tr>
<tr>
<td>Pass-through (VTAM offers support for this function, but the application program must implement the function.)</td>
</tr>
</tbody>
</table>

**Context**

Input states are not applicable to this macroinstruction.

**Syntax**

```
name

```

**Comments**

ISTGAPPC sets the following global variables:

**Table 2. LU 6.2 global macro variables set by ISTGAPPC**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Function Indicated</th>
<th>Support Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;ISTGA01</td>
<td>Conversations between transaction programs at the same LU</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA02</td>
<td>Delayed session allocation</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA03</td>
<td>Immediate session allocation</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA04</td>
<td>Sync point services</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA05</td>
<td>Program reconnect</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA06</td>
<td>Reserved</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA07</td>
<td>Session-level LU-LU verification</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA08</td>
<td>User identifier verification</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA09</td>
<td>Program-supplied user identifier and password</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA10</td>
<td>User identifier authorization</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA11</td>
<td>Profile verification and authorization</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA12</td>
<td>Reserved</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA13</td>
<td>Profile pass-through</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA14</td>
<td>Program-supplied profile</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA15</td>
<td>Send persistent verification</td>
<td>Pass-through</td>
</tr>
<tr>
<td>Variable</td>
<td>Function Indicated</td>
<td>Support Level</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>&amp;ISTGA16</td>
<td>Receive persistent verification</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA17</td>
<td>PIP data</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA18</td>
<td>Logging of data in system log</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA19</td>
<td>Flush LU's SEND buffer</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA20</td>
<td>LUW identifier</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA21</td>
<td>Prepare to receive</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA22</td>
<td>Long locks</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA23</td>
<td>Post on receipt with wait</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA24</td>
<td>Post on receipt with test for posting</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA25</td>
<td>Receive immediate</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA26</td>
<td>Test for request-to-send received</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA27</td>
<td>Data mapping</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA28</td>
<td>FMH application program data</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA29</td>
<td>Get attributes</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA30</td>
<td>Get conversation type</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA31</td>
<td>Mapped conversation LU services component</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA32</td>
<td>CHANGE_SESSION_LIMIT verb</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA33</td>
<td>MINCONTENTION WINNERS_TARGET parameter</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA34</td>
<td>RESPONSIBLE(TARGET) parameter</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA35</td>
<td>DRAIN_TARGET(NO) parameter</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA36</td>
<td>FORCE parameter</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA37</td>
<td>ACTIVATE_SESSION verb</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA38</td>
<td>DEACTIVATE_SESSION verb</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA39</td>
<td>LU parameter verbs</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA40</td>
<td>LU-LU session limit</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA41</td>
<td>Locally-known LU names</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA42</td>
<td>Uninterpreted LU names</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA43</td>
<td>Single-session reinitiation</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA44</td>
<td>Alternate code processing</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA45</td>
<td>Maximum RU size bounds</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA46</td>
<td>Session-level mandatory cryptography</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA47</td>
<td>Contention-winner automatic-activation limit</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA48</td>
<td>Queued allocation of a contention-winner session</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA49</td>
<td>Enhanced security (SAME)</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA50</td>
<td>Session-level selective cryptography</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA51</td>
<td>Conversation group support</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Table 2. LU 6.2 global macro variables set by ISTGAPPC (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Function Indicated</th>
<th>Support Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;ISTGA52</td>
<td>ALLOCATE WHEN_SESSION_FREE verb</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA53</td>
<td>LU 6.2 full-duplex protocols</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA54</td>
<td>VTAM-to-application vector list</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA55</td>
<td>Queued RCVFMH5</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA56</td>
<td>High performance data transfer</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA57</td>
<td>APPCCMD SENDRCV</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA58</td>
<td>Intra-LU conversations</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA59</td>
<td>Password substitution</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA60</td>
<td>Extended security sense codes</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA61</td>
<td>DCE security services</td>
<td>Pass-through</td>
</tr>
</tbody>
</table>

**ISTRPL6**

**Purpose**

This macroinstruction obtains storage for the RPL extension at assembly time and initializes any fields included as parameters on the macroinstruction. Any fields without a default value and not explicitly included on the macroinstruction are set to 0.

**Context**

Input states are not applicable to this macroinstruction.

**Syntax**

\[
\text{name} \rightarrow \text{ISTRPL6} \rightarrow \\
\text{CD} = \text{IMMED}\quad 1
\]

\[
\text{CD} = \text{IMMED}\quad 2
\]

\[
\text{CD} = \text{DEFER}\quad 2
\]

\[
\text{CONMODE} = \text{CS}\quad 2
\]

\[
\text{CONMODE} = \text{BUFFCA}, \text{LLCA}, \text{SAME}\quad 2
\]
LU 6.2 macroinstruction syntax and operands 525
SENSE = 32-bit_unbind_sense_code

TYPE = PROGRAM
    SERVICE
    USER

USERFLD = 4-bytes_of_user_data

VTRINA = vector_address_field

VTRINL = vector_length_field

VTROUTA = vector_address_field

VTROUTL = vector_length_field

Notes:
1 In this macroinstruction, all operands except the first must be preceded by a comma. For example, you would code ISTRPL6 CD=DEFER,FILL=LL,LOCKS=LONG.
2 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

Input parameters
CD
Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.

CD=DEFER
Specifies that the conversation state will be PEND_SEND when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CD=IMMED
Specifies that the conversation state will be SEND when the SEND indicator of the WHATRCV field is set and none of the data indicators are set. IMMED is the default.

CONMODE
Specifies that upon completion of the APPCCMD, the conversation is to be placed in logical-record-continue-any, buffer-continue-any, or continue-specific mode. This field is labeled RPL6CMOD in the RPL extension.
CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that this conversation is to apply when APPCCMD CONTROL=RECEIVE, QUALIFY=ANY is issued and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that data is to be received from this conversation by the application program only if the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that this conversation is to apply when APPCCMD CONTROL=RECEIVE, QUALIFY=ANY is issued and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

CONMODE=SAME
Specifies that the continuation mode of the conversation should remain unchanged after the completion of the APPCCMD macroinstruction using this RPL.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can only be received by a specific-type of macroinstruction for such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can only be received by either a specific-type of macroinstruction, for example, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC, or by any type of macroinstruction, for example, APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY or IANY.

CONXMOD=SAME
Specifies that the conversation mode for expedited data is to remain unchanged at the completion of this macroinstruction.

FILL
Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter corresponds to FILL=LL|BUFFER described in the LU 6.2 architecture. The field is ignored unless QUALIFY=SPEC. This field is labeled RPL6FILL in the RPL extension.

FILL=BUFF
Specifies the application program is to receive data independently of its logical-record format, up to the length specified by the AREALEN field of the RPL. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

FILL=LL
Specifies the application program is to receive one logical record, or a portion of the logical record up to the length specified by the AREALEN field of the RPL. If only a portion of the logical record is received, the DATA_INCOMPLETE bit in the what-received field is set on. The remainder of the logical record is buffered by VTAM, and will be used to satisfy the next RECEIVE request. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.
LIST
Specifies the amount of detail to be provided about LUs, modes, and sessions. The requested information is provided in a RESTORE structure and describes the LUs, modes, and sessions that have been restored. This field is labeled RPL6LIST in the RPL extension.

LIST=ALL
Specifies that all LU, mode, and session information is included in the RESTORE structure.

LIST=NONE
Specifies that no RESTORE structure is returned.

LIST=NOSESS
Specifies that all LU and mode information is included in the RESTORE structure; session information is not included.

LOCKS
Specifies when the execution of the macroinstruction is complete following execution of the CONFIRM function. This field corresponds to the LOCKS parameter on the PREPARE_TO_RECEIVE verb as described in the LU 6.2 architecture. This field is labeled RPL6LOCK in the RPL extension.

LOCKS=SHORT
Specifies that the function of this macroinstruction is complete when a positive response is received to the confirmation request.

LOCKS=LONG
Specifies that the function of this macroinstruction is complete when information, such as data, is received from the partner LU after an affirmative reply to the confirmation request. The application program must issue an APPCCMD CONTROL=RECEIVE in order to get the information that caused the prior macroinstruction to complete.

LOGMODE=8-byte_logon_mode_name
The field that holds the logon mode name of the session over which an FMH-5 flows. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN
Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

LUAFFIN=APPL
The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL
VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

LUNAME=8-byte_lu_name
The field that holds the name of a partner LU. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier
The field that holds the network identifier of the partner LU. This identifier is the network identifier of the partner LU. If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified of the target LU. (If NETID is specified, LUNAME is specified.)

The network identifier is an 8-byte name, padded on the right with blanks. This field is labeled RPL6NET in the RPL extension.
QUALIFY=one_of_the_qualify_values_listed_below
Specifies the exact function of an APPCCMD macroinstruction. The general function of the macroinstruction is determined by the CONTROL keyword, required on each APPCCMD macroinstruction.

See the individual macroinstruction descriptions for details.

ABNDPROG
Specifies abnormal termination of a conversation because of a transaction program error.

ABNDSERV
Specifies abnormal termination of a conversation because of an LU services component error.

ABNDTIME
Specifies abnormal termination of a conversation because of excessive elapsed time.

ABNDUSER
Specifies abnormal termination of a conversation because of a user-specified condition.

ACTSESS
Responds positively to a session-initiation request being processed in the LOGON or SCIP exit.

ALL
Specifies a TESTSTAT that can return status on information that is available on any conversation.

ALLOC
Allocates a session for use by a conversation.

ANY
Used to specify a RECEIVE or RCVEXPD that will accept normal or expedited information, respectively, for more than one conversation.

CNOS
Regulates session limits with another application program.

CONFIRM
Sends a confirmation request to another application program.

CONFRMD
Sends a reply to a confirmation request.

CONV
Deallocates the conversation and its underlying session.

CONVGRP
Associates a session having a specified conversation group identifier with a conversation for allocation of a conversation or deactivation of the session.

CONWIN
Allocates a conversation to a contention-winner session.

DACTSESS
Responds negatively to a session-initiation request in the LOGON or SCIP exit.

DATA
Sends data to a partner LU.

DATACON
Sends data and a confirmation request to a partner LU.

DATAFLU
Sends data to a partner LU and forces flushing of the SEND buffer.

DATAQUE
Specifies that the macroinstruction be queued pending receipt of the FMH-5 from the partner LU and that the FMH-5 as well as any data should be received to the application's buffer when received by VTAM.

DEFINE
Alters information in the LU-mode table.
DISPLAY
  Displays information in the LU-mode table.

ERROR
  Sends an error indication to a partner LU.

FLUSH
  Forces flushing of the SEND buffer.

IALL
  Specifies a TESTSTAT that can return status on information that is immediately available on any conversation.

IANY
  Specifies a RECEIVE or RCVEXPD that can receive normal or expedited information, respectively, that is immediately available from a conversation in continue-any mode.

IMMED
  Allocates a contention-winner session for immediate use by a conversation.

ISPEC
  Specifies a RECEIVE that will accept normal information that is immediately available from a user-specified conversation.

NULL
  Optional value that can be used when no other QUALIFY value applies

QUEUE
  Specifies that the macroinstruction be queued pending receipt of the FMH-5 from the partner LU and that the FMH-5 should be received to the application's buffer when received by VTAM.

RESTORE
  Restores a mode (or modes) that has been retained pending recovery of one or more persistent LU-LU sessions.

RESUME
  Releases a session that has been suspended.

RQSEND
  Requests that an application program be placed in SEND state.

SESSION
  Deactivates the session and deallocates any conversation associated with it.

SPEC
  Satisfies a RECEIVE using data for a particular conversation.

SUSPEND
  Suspends a subsequent conversation.

SYNCBEG
  Indicates the beginning of a synchronization exchange.

SYNCEND
  Indicates the end of a synchronization exchange.

WHENFREE
  Specifies to allocate a session for the conversation if a session is available or pending or one can be activated.

RTSRTRN
  Specifies, upon completion of the APPCCMD, the manner in which Request_To_Send_Received indication is to be received.

RTSRTRN=BOTh
  Specifies that Request_To_Send_Received indication can be received either by an APPCCMD CONTROL=SENDEXP or an APPCCMD CONTROL=RCVEXPD or reported in the SIGRCV and SIGDATA fields returned with other APPCCMDs.
Specifies that Request_To_Send_Received indication can be received only by an APPCCMD
CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

Specifies that the Request_To_Send_Received indication can be received only by an
APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

SENSE=32-bit_unbind_sense_code
SENSE=(32-bit_unbind_sense_code_register)
The field that holds a 32-bit sense code. This field is labeled RPL6SNSO in the RPL extension.

TYPE
Specifies the level of error being reported on an APPCCMD CONTROL=SEND, QUALIFY=ERROR macroinstruction. This field is intended to distinguish between errors to be reported to end-user transaction programs and errors to be reported to a service component, such as a mapped conversation component, of the LU. This field is labeled RPL6TYPE in the RPL extension. See “APPCCMD CONTROL=SEND, QUALIFY=ERROR” on page 427 for more details.

TYPE=PROGRAM
Specifies an end-user transaction program error is being reported.

TYPE=SERVICE
Specifies a service-component error is being reported.

TYPE=USER
Specifies that the application program is providing to VTAM a user-specific sense code that it requests be placed in the FMH-7 that VTAM creates as a result of this APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data that the application program requests be associated with a conversation. This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application. This parameter is ignored if one of the following items is true:

• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application. This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

VTROUTA=vector_address_field
VTROUTA=(vector_address_register)
Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.)

This field is labeled RPL6VAOA in the RPL extension.

VTROUTL=vector_length_field
VTROUTL=(vector_length_register)
Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.
Chapter 2. Return codes

VTAM passes feedback return codes to the LU 6.2 application program in a variety of ways. The principal feedback mechanism is the RCPRI and RCSEC return code fields in the RPL extension. These fields have meaning only when register 15 is set to X’00’ and register 0 is set to X’OB’. These values are also the values of the RPL's RTNCD and FDB2 fields, respectively.

For a general discussion of how register contents relate to RPL feedback fields, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

RCPRI and RCSEC codes

The RPL extension contains two fields in which return code information is passed to the application program at the completion of an APPCCMD macroinstruction execution. The two fields are RPL6RCPRI and RPL6RCSEC, and together they indicate the result of the macroinstruction execution, including any state changes to the specified conversation. The RCPRI field returns a primary return code to the application; the RCSEC field returns a secondary return code to the application. Some RCPRI codes do not have associated RCSEC subcodes. For these RCPRI codes, the RCSEC field is set to X’0000’.

Some of the (RCPRI, RCSEC) return codes indicate the results of the local VTAM’s processing of the macroinstruction; these return codes are returned on the APPCCMD that invoked the local processing. Other (RCPRI, RCSEC) return codes indicate the results of processing invoked at the remote end of the conversation and, depending upon the CONTROL and QUALIFY settings of the APPCCMD, can be returned on the APPCCMD that invoked the remote processing or on a subsequent APPCCMD. Still other return codes report events that originate at the remote end of the conversation.

The following information describes the RCPRI and RCSEC codes. Each description includes the meaning of the code, the reason for the condition indicated by the code, when the code can be reported to the application program, and the state of the conversation (if applicable) when the function of the APPCCMD completes. Actions taken by the local application program are discussed in the return code descriptions in terms of APPCCMD macroinstructions; actions taken by the remote LU or transaction program are described more generically using the architected protocol boundary verbs documented in the LU 6.2 architecture.

Note: Some application programs change the hexadecimal values from the RCPRI, RCSEC fields to decimal values. You may need to convert these back to hexadecimal values for problem determination.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>(all)</td>
<td>USF6OK</td>
<td>OK</td>
</tr>
</tbody>
</table>

The local application program issued an APPCCMD macroinstruction that executed without error. The function defined for the APPCCMD was performed as specified.

The OK RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the application; the RCSEC subcode provides additional information.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>USF6OKSC</td>
<td>OK</td>
</tr>
</tbody>
</table>

The APPCCMD completed successfully and no additional information is defined for the APPCCMD. If a conversation-related macroinstruction is issued, the conversation state can be found in the CONSTATE field. Whenever this RCPRI,RCSEC combination is present, registers 15 and 0 are also set to 0.
The CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction were accepted by the partner LU as specified, without negotiation.

One or more of the CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was changed by negotiation with the partner LU. The values are returned to the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction. (The macroinstruction description defines which values can be negotiated.)

VTAM has determined that the partner LU supports only single sessions. If the session limit you specified was greater than 1, or if you did not specify a session limit, then the default values of 1, 0, 0 were used for your CNOS request.

If the partner LU indicated single-session capability using a negative BIND response, the partner LU's name will be missing from the Userdata subfield of the BIND. When the application program issues an APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY macroinstruction, it should verify the presence of the partner LU's fully qualified name. If the FQNLEN field is 0, the partner LU's name is not available. Check the FQNLEN field before checking the FQNAME field.

VTAM rejected the APPCCMD CONTROL=REJECT, QUALIFY=SESSION macroinstruction because of an internal error other than a storage shortage condition.
The APPCCMD CONTROL=OPRCNTL,QUALIFY=RESTORE macroinstruction is unnecessary. The associated mode (or modes) has been restored already, or nothing existed to restore.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0007'</td>
<td>USF6RSIN</td>
<td>RESTORE_INCOMPLETE—INPUT_WORK_AREA_TOO_SMALL</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL,QUALIFY=RESTORE macroinstruction is incomplete. The AREA supplied is too small to hold all the information that needs to be returned. Reissue the macroinstruction one or more times to obtain all the restore information and to complete the restore.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>USF6NINA</td>
<td>NO IMMEDIATELY AVAILABLE INFORMATION</td>
</tr>
</tbody>
</table>

An APPCCMD that requested the immediate return of available information was issued. However, no information that could satisfy the request was available.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>USF6RTEC</td>
<td>REQUEST TERMINATED BY END OF CONVERSATION</td>
</tr>
</tbody>
</table>

An APPCCMD was awaiting processing or awaiting the arrival of information or a response on a specific conversation. The command has terminated because the conversation ended before the requested information became available or before it could be processed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>USF6ANMS</td>
<td>SESSIONS WILL USE APPL NAME, GENERIC NAME REQUESTED</td>
</tr>
</tbody>
</table>

Use of the generic resource name was requested but the application network name is required.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>USF6GNMS</td>
<td>SESSIONS WILL USE GENERIC NAME, APPL NAME WAS REQUESTED</td>
</tr>
</tbody>
</table>

Use of the application network name was requested but the generic resource name is required.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000C'</td>
<td>USF6NAM1</td>
<td>AS SPECIFIED, PARTNER LU KNOWN BY DIFFERENT NAME</td>
</tr>
</tbody>
</table>

The CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction were acceptable by the partner LU as specified, without negotiation. Furthermore, the CNOS operation caused an LU entry of type RCVD_NAME to be changed to a VARIANT_NAME entry in the LU-mode table.
### Table: RCPRI, RCSEC, ISTUSFBC EQU

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000D'</td>
<td>USF6NAM2</td>
<td>AS NEGOTIATED, PARTNER LU KNOWN BY DIFFERENT NAME</td>
</tr>
</tbody>
</table>

One or more of the CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was changed by negotiation with the partner LU. The values are returned to the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction. (The macroinstruction description defines which values can be negotiated.) Furthermore, the CNOS operation caused an LU entry of type RCVD_NAME to be changed to a VARIANT_NAME entry in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>(all)</td>
<td>USF6ALLC</td>
<td>ALLOCATION ERROR</td>
</tr>
</tbody>
</table>

The application program issued APPCCMD CONTROL=ALLOC and allocation of the specified conversation could not be completed. When the ALLOCATION_ERROR RCPRI code is used with one of the RCSEC subcodes (X'0000'–X'000F'), they form the complete return code that is returned to the program. The RCSEC subcode identifies the specific error. (The partner LU and remote transaction program referred to in the RCSEC definitions are the LU named in the LUNAME field of the APPCCMD, and the transaction program named in the FMH-5 supplied through the AREA field of the APPCCMD, respectively.)

If the partner LU detects the error that causes an ALLOCATION_ERROR RCPRI code to be returned to the application, the error indicator sent by the partner LU can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction. If an ALLOCATION_ERROR RCPRI code is returned to the application along with LOGRCV=YES, the conversation should issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to receive the error log data. When the error log data is received, the conversation is over.

If an ALLOCATION_ERROR RCPRI code is returned to the application along with LOGRCV=NO, the conversation is in END_CONV state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>USF6ALNR</td>
<td>ALLOCATION FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>

The conversation cannot be allocated on a session because of a permanent condition. For example, the session to be used for the conversation cannot be activated for one of the reasons:

- The mode is closed; the current session limit is 0.
  - CNOS has not been negotiated and no entry has been created for the mode.
  - A previous CNOS request has set limits to 0.
- A system definition error.
- A session-activation protocol error.

The session also might be deactivated because of a session protocol error before the conversation could be allocated. The application program should not try the allocation request again until the condition is corrected. The application should check the returned SENSE field in the RPL extension for an indication of the exact error.

If this code occurs when issuing a DISPLAY APING operator command, the session may have been deactivated as a result of processing a received APING request for the same mode. Reissue the operator command.
<table>
<thead>
<tr>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>USF6ALR</td>
<td>ALLOCATION FAILURE RETRY</td>
</tr>
<tr>
<td>USF6ALCM</td>
<td>CONVERSATION TYPE MISMATCH</td>
</tr>
<tr>
<td>USF6ALPI</td>
<td>PIP NOT ALLOWED</td>
</tr>
<tr>
<td>USF6ALPP</td>
<td>PIP NOT SPECIFIED CORRECTLY</td>
</tr>
<tr>
<td>USF6ALSC</td>
<td>SECURITY NOT VALID</td>
</tr>
<tr>
<td>USF6ALSY</td>
<td>SYNC LEVEL NOT SUPPORTED BY LU</td>
</tr>
</tbody>
</table>

The conversation cannot be allocated on a session because of a temporary condition. For example, the session to be used for the conversation cannot be activated because of a temporary lack of resources at the remote LU; or the session was deactivated because of session outage before the conversation could be allocated. The condition is temporary, and the program can try the allocation request again.

The partner LU rejected the allocation request because the remote transaction program does not support the respective mapped or basic protocol boundary. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the local application program provided program initialization parameter (PIP) data (along with the FMH-5) and either the partner LU does not support PIP data, or the remote transaction program has no PIP variables defined. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the remote transaction program has one or more PIP variables defined and the local application program provided no program initialization parameters, or the local application program specified program initialization parameters (along with the FMH-5) that do not correspond in number to those defined for the remote transaction program. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the access security information supplied by the local application (in the FMH-5) is not valid. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the synchronization level specified in the allocation request is not supported by both the local and partner LU. The local LU specifies its level of synchronization support on its APPL statement. The partner LU has returned the negotiated level between the two LUs in the BIND response. This return code is returned on the APPCCMD CONTROL=ALLOC macroinstruction for the local LU.
The partner LU rejected the allocation request because the local application program specified a synchronization level (in the FMH-5) that the remote transaction program does not support. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the local application program specified a remote transaction program name (TPN) that the partner LU does not recognize. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the local application program specified a remote transaction program that the partner LU recognizes but cannot start. The condition is not temporary, and the application should not try the allocation request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the local application specified a remote program that the remote LU recognizes but currently cannot start. The condition is temporary, and the application can try the allocation request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the reconnection request because it does not recognize the conversation correlator. The condition is not temporary, and the application should not try the reconnection request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the reconnection request because it currently cannot reconnect the remote transaction program implied by the conversation correlator. The condition is temporary, however, and the application can try the reconnection request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

<table>
<thead>
<tr>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0007' USF6ALSL SYNC LEVEL NOT SUPPORTED BY PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008' USF6ALTP TPN NOT RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009' USF6ALTN TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A' USF6ALTR TRANSACTION PROGRAM NOT AVAILABLE, RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B' USF6ALRN CANNOT RECONNECT TRANSACTION PROGRAM, NO RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C' USF6ALRR CANNOT RECONNECT TRANSACTION PROGRAM, RETRY</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
</tr>
</tbody>
</table>

The partner LU rejected the allocation request because the local application program specified a recovery level of program reconnect (in the FMH-5) and the remote transaction program does not support program reconnect. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>USF6SPMA</td>
<td>MODE MUST BE RESTORED BEFORE USING</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=ALLOC macroinstruction is rejected because the specified mode name is pending recovery for persistent LU-LU sessions. Restore the mode by issuing APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>USF6DARQ</td>
<td>DEALLOCATION REQUESTED</td>
</tr>
</tbody>
</table>

The allocation request has been canceled before its normal processing could be completed. The local application program issued a request for abnormal deallocation of the pending conversation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>USF6ALSF</td>
<td>ALLOCATION ERROR - SYNCH LEVEL NOT VALID FOR FULL-DUPLEX</td>
</tr>
</tbody>
</table>

The allocation request has been rejected because it specifies a full-duplex conversation with a sync point level not allowed for a full-duplex conversation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>USF6LNSF</td>
<td>ALLOCATION ERROR - LU PAIR NOT SUPPORTING FDX CONVERSATION</td>
</tr>
</tbody>
</table>

The allocation request has been rejected because it specifies a full-duplex conversation and the negotiated level of support between the local application and the partner LU does not allow full-duplex conversations.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>(all)</td>
<td>USF6CNSA</td>
<td>CNOS FAILURE</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not process successfully. The CNOS_ALLOCATION_ERROR RCPRI code together with one of the RCSEC subcodes (X'0000'–X'0006') form the complete return code that is returned to the transaction program. The RCSEC subcode identifies the specific error. The local and partner LUs' CNOS parameters are not changed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0000'</td>
<td>USF6CANR</td>
<td>ALLOCATION FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>
The control operator conversation cannot be allocated because of a condition that is not temporary. For example, the session to be used for the control operator conversation cannot be activated because the session limit for the specified partner LU and SNASVCMG mode name is currently 0 at either the local LU or partner LU; or because of a system definition error or a session-activation protocol error; or because a session protocol error caused the session to be deactivated before the conversation could be allocated. The CNOS will not be able to complete successfully until the condition is corrected. This code can also be returned if a partner LU rejects a SNASVCMG mode name BIND.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0008’</td>
<td>X’0001’</td>
<td>USF6CAR</td>
<td>ALLOCATION FAILURE, RETRY</td>
</tr>
</tbody>
</table>

The control operator conversation cannot be allocated because of a temporary condition. For example, the session to be used for the control operator conversation cannot be activated because of a temporary lack of resources at the local LU or partner LU, or the session was deactivated because of session outage before the conversation could be allocated. The condition is temporary, and the control operator can try the transaction again later.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0008’</td>
<td>X’0002’</td>
<td>USF6CATR</td>
<td>TRANSACTION PROGRAM NOT AVAILABLE, RETRY</td>
</tr>
</tbody>
</table>

The partner LU is currently unable to start the transaction program identified as hex 06F1, which is the SNA service transaction program for the control operator. For example, there can be a temporary lack of resources the partner LU needs to start the transaction program. The condition is temporary, and the control operator can try the transaction again later.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0008’</td>
<td>X’0003’</td>
<td>USF6CATN</td>
<td>TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY</td>
</tr>
</tbody>
</table>

The partner LU is unable to start the transaction program identified as X’06F1’, which is the SNA service transaction program for the control operator. The condition is not temporary, and the application should not try again the CNOS request.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0008’</td>
<td>X’0004’</td>
<td>USF6CACM</td>
<td>CONVERSATION TYPE MISMATCH</td>
</tr>
</tbody>
</table>

The partner LU rejected the CNOS conversation allocation request because the remote transaction program does not support the respective mapped or basic protocol boundary.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0008’</td>
<td>X’0005’</td>
<td>USF6CASC</td>
<td>SECURITY NOT VALID</td>
</tr>
</tbody>
</table>

The partner LU rejected the CNOS conversation allocation request because the access security information supplied by VTAM (in the FMH-5) is not valid.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0008’</td>
<td>X’0006’</td>
<td>USF6SPMC</td>
<td>MODE MUST BE RESTORED BEFORE USING</td>
</tr>
</tbody>
</table>
The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction is rejected because the specified mode name is pending recovery for persistent LU-LU sessions. Restore the mode by issuing APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE. New modes can be added once the SNASVCMG mode for an LU has been restored, but any mode that exists when the failure (or takeover) occurs cannot be used until that mode has been restored.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQUI</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0008’</td>
<td>X’0007’</td>
<td>USF6NQNM</td>
<td>NETWORK QUALIFIED NAME MISMATCH</td>
</tr>
</tbody>
</table>

The name on an APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was an ACB name. The ACB name is not identical to the network resource name. ACB names cannot be used in cross-domain, cross-network, or network qualified. For information on coding the ACBNAME operand, see the z/OS Communications Server: SNA Resource Definition Reference.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQUI</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’000C’</td>
<td>X’0000’</td>
<td>USF6CNSN</td>
<td>CNOS RESOURCE FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because of a failure that caused the control operator conversation to be deallocated prematurely. For example, the session being used for the control operator conversation was deactivated for one of the reasons:

- A session protocol error
- A session outage from which the control operator component of the LU could not recover

The conversation also might be deallocated because of a protocol error between the control operator components of the LUs. The condition is not temporary, and the control operator should not try the transaction again until the condition is corrected. The CNOS parameters remain unchanged at the local LU, or both the local and partner LUs, depending on when the failure occurred.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0010’</td>
<td>(all)</td>
<td>USF6CRRJ</td>
<td>COMMAND RACE REJECT</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because two CNOS operations caused contention for the needed resources.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0010’</td>
<td>X’0000’</td>
<td>USF6CRPR</td>
<td>PARTNER GRANTED RETRY</td>
</tr>
</tbody>
</table>

Both LUs initiated a CNOS negotiation for the same mode at the same time. The partner LU will try the CNOS request again. VTAM fails the CNOS request from the local LU.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0010’</td>
<td>X’0001’</td>
<td>USF6CRLR</td>
<td>CONTROL OPERATOR FOR LOCAL LU RETRIED</td>
</tr>
</tbody>
</table>

Both LUs initiated CNOS processing for the same mode at the same time. VTAM failed the partner’s CNOS attempt, and the local LU was given permission to try the CNOS request again. VTAM attempted CNOS processing again but the subsequent CNOS negotiation failed as well. VTAM was forced to fail the local LU’s CNOS request.
<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0010'</td>
<td>X'0002'</td>
<td>USF6PCIP</td>
<td>PARTNER CNOS IN PROGRESS</td>
</tr>
</tbody>
</table>

The partner LU has already begun processing a CNOS for the same mode name, and its processing will continue uninterrupted. The application program must reissue this APPCCMD for it to be processed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0010'</td>
<td>X'0003'</td>
<td>USF6LPSS</td>
<td>LU IN PENDING SINGLE STATE</td>
</tr>
</tbody>
</table>

The CNOS negotiation cannot be attempted at this time because the partner LU has initiated a CNOS request for the same mode. The partner LU might be a single-session-capable LU. The local LU cannot issue a CNOS request until the CNOS request initiated by the partner LU completes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0010'</td>
<td>X'0004'</td>
<td>USF6PLSS</td>
<td>PARTNER LU STARTING SESSION</td>
</tr>
</tbody>
</table>

A partner LU that provides only single-session support is currently initiating a session. Because only one session can be active at a time, the application program's CNOS request is rejected. The application program can try the CNOS command again later.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>USF6DABP</td>
<td>DEALLOCATE ABEND PROGRAM</td>
</tr>
</tbody>
</table>

The remote transaction program issued a DEALLOCATE verb, as defined in the LU 6.2 architecture, specifying the TYPE(ABEND_PROG) parameter, or the remote LU did so because of a remote transaction program abend condition. If the conversation for the remote transaction program was in a state in which information can be received when the DEALLOCATE was issued, information sent by the local application and not yet received by the remote transaction program was purged. This return code can be reported to the local application on any APPCCMD macroinstruction that can process the error notification on a half-duplex conversation. This return code can only be reported on an APPCCMD CONTROL=RECEIVE on a full-duplex conversation. The error indicator sent by the partner LU to specify the DEALLOCATE_ABEND_PROGRAM condition can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction. If a DEALLOCATE_ABEND_PROGRAM RCPRI code is returned to the application along with LOGRCV=YES, the conversation should issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to receive the error log data. The conversation is then ended. If a DEALLOCATE_ABEND_PROGRAM RCPRI code is returned to the application along with LOGRCV=NO, the conversation is ended.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>USF6DABS</td>
<td>DEALLOCATE ABEND SERVICE</td>
</tr>
</tbody>
</table>

The remote transaction program issued a DEALLOCATE verb, as described in the LU 6.2 architecture, specifying the TYPE(ABEND_SVC) parameter. If the conversation for the remote transaction program was in a state in which information can be received when the DEALLOCATE was issued, information sent by the local application and not yet received by the remote transaction program was purged. This return code can be reported to the local application on any APPCCMD macroinstruction that can process the error notification on a half-duplex conversation. This return code can only be reported on an APPCCMD CONTROL=RECEIVE on a full-duplex conversation. The error indicator sent by the partner LU to specify the DEALLOCATE_ABEND_SERVICE condition can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction.
macroinstruction. If a DEALLOCATE_ABEND_SERVICE RCPRI code is returned to the application along with LOGRCV=YES, the conversation is in PEND_END_CONV_LOG or PEND_RESET_LOG state. If a DEALLOCATE_ABEND_SERVICE RCPRI code is returned to the application along with LOGRCV=NO, the conversation is in END_CONV or FDX_RESET state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>USF6DABT</td>
<td>DEALLOCATE ABEND TIMER</td>
</tr>
</tbody>
</table>

The remote transaction program issued a DEALLOCATE verb, as described in the LU 6.2 architecture, specifying the TYPE(ABEND_TIMER) parameter. If the conversation for the remote program was in a state in which information can be received when the DEALLOCATE was issued, information sent by the local application program and not yet received by the remote transaction program was purged. This return code can be reported to the local program on any APPCCMD macroinstruction that can process the error notification on a half-duplex conversation. This return code can only be reported on an APPCCMD CONTROL=RECEIVE on a full-duplex conversation. The error indicator sent by the partner LU to specify the DEALLOCATE_ABEND_TIMER condition can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction. If a DEALLOCATE_ABEND_TIMER RCPRI code is returned to the application along with LOGRCV=YES, the conversation is in PEND_END_CONV_LOG or PEND_RESET_LOG state. If a DEALLOCATE_ABEND_TIMER RCPRI code is returned to the application along with LOGRCV=NO, the conversation is in END_CONV or FDX_RESET state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0020'</td>
<td>X'0000'</td>
<td>USF6CNSR</td>
<td>CNOS FAILURE, RETRY</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued and a conversation was begun with the partner LU. However, a failure occurred that caused the conversation to be prematurely terminated. For example, the session being used for the conversation was deactivated because of a session outage, such as a line failure or a modem failure. The condition is temporary, and the application can try the transaction again.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>USF6LRBE</td>
<td>LOGICAL RECORD BOUNDARY ERROR</td>
</tr>
</tbody>
</table>

The application program began sending a logical record before the previous logical record was sent in its entirety. The conversation state does not change.

For macroinstructions that use the QUALIFY=DATACON keyword, the data that was to be sent with the confirmation request is held. The application program must either furnish more data to finish the logical record, or truncate the incomplete record. The application cannot immediately send more data to complete the logical record, but must explicitly flush the send buffer and then send data to complete the logical record.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0028'</td>
<td>X'0000'</td>
<td>USF6SLCL</td>
<td>LU MODE SESSION LIMIT CLOSED</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because the partner LU currently will not allow the session limit for the specified mode name to be raised above 0. The session limit remains at 0. This condition is not necessarily permanent; the control operator can try the CNOS transaction again later.
VTAM rejected the APPCCMD because one of the RPL, RPL extension, or session limits structure fields specified in the APPCCMD contained a value that was not valid. The PARAMETER_ERROR RCPRI code together with the RCSEC subcodes (X'0000'–X'002D') form the complete return code that is returned to the application. The subcode identifies the specific error. This RCPRI code is returned on the APPCCMD that contained the parameter that was not valid. When this RCPRI code is returned on a conversation APPCCMD macroinstruction (that is, a macroinstruction that does not specify CONTROL=OPRCNTL), the state of the conversation remains unchanged. When this RCPRI code is returned on an APPCCMD CONTROL=OPRCNTL macroinstruction, the local and partner LUs' CNOS parameters are not changed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>(all)</td>
<td>USF6Parm</td>
<td>PARAMETER ERROR</td>
</tr>
</tbody>
</table>

The APPCCMD specified an unrecognized partner LU name or network identifier.

This combination of return codes might result if VTAM does not find the LU name for a partner in the LU-mode table. The partner LU name and the logon mode name are added to the dynamically built LU-mode table during CNOS negotiation. To initiate CNOS negotiation, the application program issues the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction and specifies the LU name and logon mode (LOGMODE) name to be used during communication.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>USF6IVLU</td>
<td>INVALID LU NAME OR NETWORK IDENTIFIER</td>
</tr>
</tbody>
</table>

The APPCCMD specified an unrecognized logmode name, or the logmode name is not allowed for the LU-LU pair.

This combination of return codes might occur if the LU name specified for a conversation allocation request is present in the LU-mode table but the logon mode name is not present. The partner LU name and the logon mode name are added to the dynamically built LU-mode table during CNOS negotiation. To initiate CNOS negotiation, the application program issues the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction and specifies the LU name and logon mode (LOGMODE) name to be used during communication.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>USF6IVMD</td>
<td>INVALID MODE</td>
</tr>
</tbody>
</table>

The APPCCMD specified an unassigned conversation ID, or the RPL used for the request specified an ACB other than the one associated with the conversation assigned that CONVID. The value specified might have been a valid CONVID, but the conversation might not be active.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>USF6IVCI</td>
<td>INVALID CONVERSATION</td>
</tr>
</tbody>
</table>

The data provided by the application program on an APPCCMD CONTROL=SEND, an APPCCMD CONTROL=PREPRCV, or an APPCCMD CONTROL=DEALLOC macroinstruction was not valid. It contained a logical record length (LL) value of X'0000', X'0001', X'8000', or X'8001'. An LL value of hex 0001, which indicates that the data contains a presentation services (PS) header for sync point, is allowed only on conversations with a synchronization level of sync point.
### Return Codes

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0004’</td>
<td>USF6IVSV</td>
<td>INVALID VALUES FOR SNASVCMG MODE</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued and the values specified for the SESSLIM, MINWINL, and MINWINR do not specify (2,1,1) or (0,0,0), respectively.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0005’</td>
<td>USF6IVDL</td>
<td>INVALID DRAINL CHANGE</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued, NBRMODE=ONE and DRAINL=YES were specified, the session limit in effect when the APPCCMD was issued was 0, and DRAINL=NO was in effect when the APPCCMD was issued. (The application program attempted to change DRAINL from NO to YES on an APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction when session limits were 0.)

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0006’</td>
<td>USF6SNAR</td>
<td>SNASVCMG MODE CANNOT CURRENTLY BE RESET</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction is issued, the SNASVCMG mode name is specified, and either one or more session limits for the mode name group for the partner LU is not 0; or one or more session limits for the mode name group for the partner LU are 0, but draining is enabled.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0007’</td>
<td>USF6MMEX</td>
<td>MINWINL PLUS MINWINR EXCEEDS SESSLIM</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS or QUALIFY=DEFINE macroinstruction was issued and either the sum of MINWINL plus MINWINR is greater than the SESSLIM value specified, or the sum of DMINWNL plus DMINWNR is greater than the DSESSLIM value specified.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0008’</td>
<td>USF6LNIN</td>
<td>SUPPLIED LENGTH INSUFFICIENT</td>
</tr>
</tbody>
</table>

The application issued one of the macroinstructions:
- APPCCMD CONTROL=RCVEXPD
- APPCCMD CONTROL=RCVFHM5
- APPCCMD CONTROL=RECEIVE,OPTCD=XBUFLST
- APPCCMD CONTROL=OPRCNTL,QUALIFY=ACTSESS
- APPCCMD CONTROL=OPRCNTL,QUALIFY=DISPLAY
- APPCCMD CONTROL=OPRCNTL,QUALIFY=RESTORE
- APPCCMD CONTROL=TESTSTAT.

The data area or data length was not suitable as indicated in the items:

**RECEIVE,OPTCD=XBUFLST**

The area specified is not large enough to hold one extended buffer list entry.

**RCVEXPD**

Data area is too small to contain all the expedited data.

Return codes 545
RCVFHM5
Data area is too small to contain the next available FMH-5.

QUALIFY=ACTSESS
Data length indicated in the supplied session parameters was larger than the amount of data provided or exceeds the maximum size allowed.

QUALIFY=DISPLAY
Data area is too small to contain the DEFINE/DISPLAY (ISTSLD) structure.

QUALIFY=RESTORE
Data area is too small to contain the RESTORE (ISTREST) structure.

TESTSTAT
Data area is too small to contain the status data structure (ISTSTATD).

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSE</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0009’</td>
<td>USF6INSL</td>
<td>INCOMPLETE STRUCTURE SUPPLIED</td>
</tr>
</tbody>
</table>

The application program issued one of the macroinstructions:
- APPCCMD CONTROL=OPRCNTRL, QUALIFY=ACTSESS
- APPCCMD CONTROL=OPRCNTRL, QUALIFY=CNOS
- APPCCMD CONTROL=OPRCNTRL, QUALIFY=DEFINE.

The data length was not suitable as indicated in the:

QUALIFY=ACTSESS
Data length provided was less than the minimum size for the session parameters.

QUALIFY=CNOS
Data length provided was less than the minimum size for the session limits structure (ISTSLCNS).

QUALIFY=DEFINE
Data length provided was less than the minimum size for the DEFINE/DISPLAY (ISTSLD) structure.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSE</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’000A’</td>
<td>USF6INFM</td>
<td>INCOMPLETE FMH5 SUPPLIED</td>
</tr>
</tbody>
</table>

The application program issued APPCCMD CONTROL=ALLOC, but did not supply an entire FMH-5.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSE</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’000B’</td>
<td>USF6INGD</td>
<td>INCOMPLETE GDS VARIABLE SUPPLIED</td>
</tr>
</tbody>
</table>

The application program issued an abnormal termination APPCCMD deallocation macroinstruction, but did not supply an entire GDS variable.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSE</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>USF60EXT</td>
<td>ZERO EXIT FIELD</td>
</tr>
</tbody>
</table>

The RPL specified that the ECB-EXIT field is being used as an EXIT field, but the RPL exit routine address in the field is 0. No RPL exit routine has been scheduled.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSE</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>USF60ECB</td>
<td>ZERO ECB FIELD</td>
</tr>
</tbody>
</table>
The RPL specified that the ECB-EXIT field is being used to point to an external ECB, but the address in the field is 0. No ECB has been posted.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>USF6RIAS</td>
<td>REQUEST INVALID FOR ADDRESS SPACE</td>
</tr>
</tbody>
</table>

An internal error occurred.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>USF6CBIN</td>
<td>CONTROL BLOCK INVALID</td>
</tr>
</tbody>
</table>

The RPL's ACB field does not contain the address of a valid ACB or the ACB is closed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>USF6INDL</td>
<td>INVALID DATA ADDRESS OR LENGTH</td>
</tr>
</tbody>
</table>

An APPCCMD was issued that specified a work area address that is beyond the addressable range of the application program.

If using a buffer list or extended buffer list to send data, check entries to ensure that the length field does not contain any negative values.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>USF6PRVO</td>
<td>PREVIOUS MACROINSTRUCTION OUTSTANDING</td>
</tr>
</tbody>
</table>

An APPCCMD is issued that specifies a conversation resource while an outstanding macroinstruction that targets the same conversation and processes on the same conversation queue is pending completion, or an APPCCMD CONTROL=OPRCNTL is issued while an outstanding operator control APPCCMD that targets the same LU is pending completion. Wait until the first macroinstruction completes or coordinate this request with the one that is outstanding.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>USF6BLIV</td>
<td>BUFFER LIST LENGTH INVALID</td>
</tr>
</tbody>
</table>

The RECLLEN field of the RPL was not valid.

For the macroinstructions, the RECLLEN field must be a nonzero multiple of 16:

- APPCCMD CONTROL=DEALLOC, OPTCD=BUFFLST
- APPCCMD CONTROL=PREPRCV, OPTCD=BUFFLST
- APPCCMD CONTROL=SEND, OPTCD=BUFFLST
- APPCCMD CONTROL=SENDEXPD, OPTCD=BUFFLST
- APPCCMD CONTROL=SENDRCV, OPTCD=BUFFLST.

For the macroinstructions, the RECLLEN field must be a nonzero multiple of 48:

- APPCCMD CONTROL=DEALLOC, OPTCD=XBUFLST
- APPCCMD CONTROL=PREPRCV, OPTCD=XBUFLST
- APPCCMD CONTROL=SEND, OPTCD=XBUFLST.
For the APPCCMD CONTROL=SENDRCV, OPTCD=XBUFLST macroinstruction, the value for RECLEN minus 16 must be a nonzero multiple of 48.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0013'</td>
<td>USF6NOMD</td>
<td>NO CORRESPONDING MODE IN LM TABLE</td>
</tr>
</tbody>
</table>

The application program issued one of the macroinstructions:

- APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY
- APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE.

The application program also specified a mode name for which no corresponding entry exists in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0014'</td>
<td>USF6IVBP</td>
<td>INVALID BIND PARAMETERS</td>
</tr>
</tbody>
</table>

The application program issued an APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS and specified a set of BIND parameters that were not valid, or the parameters in the BIND that was received were not valid.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>USF6IVTP</td>
<td>INVALID TPN</td>
</tr>
</tbody>
</table>

The application program issued an APPCCMD CONTROL=ALLOC with an FMH-5 that contained a transaction program name that was reserved or not valid, such as X'06F1', which is the SNA service transaction program for the control operator.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0016'</td>
<td>USF6NOLU</td>
<td>NO CORRESPONDING LU IN LM TABLE</td>
</tr>
</tbody>
</table>

The application program issued one of the macroinstructions:

- APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY
- APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE.

The application program also specified an LU name for which no corresponding entry exists in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0017'</td>
<td>USF6IMDF</td>
<td>INVALID MODE SPECIFIED</td>
</tr>
</tbody>
</table>

The application program issued an APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE macroinstruction and specified mode name SNASVCMG.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0018'</td>
<td>USF6ILSP</td>
<td>INVALID LIMIT SPECIFIED</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued and one of the session limit fields was an incorrect value.
### Return Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>SNASVCMG MODE ALREADY INITIALIZED</td>
</tr>
<tr>
<td>X'0019'</td>
<td>SNASVCMG MODE ALREADY INITIALIZED</td>
</tr>
<tr>
<td>X'001A'</td>
<td>ALL MODES SPECIFIED ON SINGLE SESSION LU</td>
</tr>
<tr>
<td>X'001B'</td>
<td>SNASVCMG OR CPSVCMG MODE FOR SINGLE SESSION LU</td>
</tr>
<tr>
<td>X'001C'</td>
<td>SINGLE SESSION, MODE ALREADY INITIALIZED</td>
</tr>
<tr>
<td>X'001D'</td>
<td>CID INVALID</td>
</tr>
<tr>
<td>X'001E'</td>
<td>CID INVALID</td>
</tr>
<tr>
<td>X'001F'</td>
<td>APPCCMD ISSUED FOR NON-APPC</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued in order to initialize the SNASVCMG mode. However, it was already initialized, and no action was taken.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued against all the mode names of the LU specified. However, the partner LU is single-session capable. Therefore, an APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction must be issued against a specific mode name.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued for the SNASVCMG or CPSVCMG mode name. However, the partner LU is single-session capable, and the SNASVCMG or CPSVCMG is not allowed.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued for a partner LU that is single-session capable. However, another of the LU’s mode names is already initialized to nonzero session limits, and only one mode name can have nonzero session limits at a time.

The RPL’s ARG field does not contain a valid session identifier (CID). You might have inadvertently modified the field or failed to set it in the first place, or you might have used the CID of a session that no longer exists.

The application issued an APPCCMD against a non-LU 6.2 session or resource. The APPCCMD is rejected.

An APPCCMD CONTROL=REJECT request was issued. However, a previous APPCCMD CONTROL=REJECT request has already been issued for the same resource. The later APPCCMD CONTROL=REJECT was rejected.
One of the macroinstructions was issued:

- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER.

However, a prior macroinstruction that cannot be canceled is outstanding. The command is not allowed in this case and is rejected. This command also is not allowed to be issued when the conversation is in RECEIVE state and no data has been received for the conversation. APPCCMD CONTROL=REJECT, QUALIFY=CONV can be issued to terminate the conversation and session in this case.

An undefined value for the CONTROL or QUALIFY keyword was specified, or a QUALIFY value is not valid to use with the specified CONTROL value. For CONTROL types that do not use a QUALIFY value, RPL6QUAL must be set to 0.

VTAM rejected an APPCCMD CONTROL=REJECT, QUALIFY=SESSION request or an APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND request or an APPCCMD CONTROL=SETSESS, QUALIFY=RESUME request because the local application specified:

- A session instance identifier for a session that was not active at the time of the request.
- A session ID length that was not valid.

VTAM rejected the APPCCMD CONTROL=SEND request because the local application did not supply a complete PS header. (For example, the PS header length and data are missing.)

VTAM rejected the APPCCMD CONTROL=SEND request because the local application specified an insufficient PS header length (the length equals 0).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>USF6DARJ</td>
<td>ABNORMAL DEALLOCATE REJECTED, RETRY</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0022'</td>
<td>USF6IVCQ</td>
<td>INVALID CONTROL OR QUALIFY VALUE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>USF6INSI</td>
<td>INVALID SESSION INSTANCE IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>USF6PSHI</td>
<td>PS HEADER NOT SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>USF6PSLI</td>
<td>PS HEADER LENGTH IS INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0026'</td>
<td>USF6NMSC</td>
<td>SESSION INSTANCE IDENTIFIER AND CONVERSATION IDENTIFIER MISMATCH</td>
</tr>
</tbody>
</table>
VTAM rejected the APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND request because the application program requested a session with APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND, but the conversation identified by CONVID was not currently assigned to the session identified by SESSID. VTAM rejected the request and nothing was suspended.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0027'</td>
<td>USF6IDET</td>
<td>Label</td>
<td>INVALID DEACTIVATION TYPE CODE</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=REJECT, QUALIFY=SESSION request because the local application program omitted the DEACTYP parameter or specified an UNBIND deactivation type code value other than cleanup (X'0F') or protocol violation (X'FE'). The session has been successfully deactivated with UNBIND (X'0F').

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>USF6NCRY</td>
<td>Label</td>
<td>CRYPTOGRAPHY NOT ALLOWED ON MODE</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=SEND, an APPCCMD CONTROL=PREPRCV, or an APPCCMD CONTROL=DEALLOC macroinstruction is rejected because CRYPT=YES is specified, and the mode does not support encryption.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0029'</td>
<td>USF6INLI</td>
<td>Label</td>
<td>INVALID LIST VALUE SPECIFIED ON APPCCMD FOR RESTORE</td>
</tr>
</tbody>
</table>

The value for the LIST field in the RPL is not equal to NONE, ALL, or NOSESS. The keyword LIST=ALL, LIST=NONE, or LIST=NOSESS can be specified on the APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE macroinstruction.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'002A'</td>
<td>USF6INCG</td>
<td>Label</td>
<td>INVALID CGID VALUE SPECIFIED</td>
</tr>
</tbody>
</table>

A macroinstruction was issued specifying CONVGRP, but the conversation group ID (CGID) was not valid. You might have unintentionally modified the field, failed to set it correctly, or used a CGID that corresponds to a session that no longer exists.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>USF6NONI</td>
<td>Label</td>
<td>NETWORK-QUALIFIED NAME REQUIRED</td>
</tr>
</tbody>
</table>

NETID was not coded on the APPCCMD although PARMS=(NQNAMES=YES) was coded on the ACB macroinstruction.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'002C'</td>
<td>USF6INEL</td>
<td>Label</td>
<td>PARAMETER ERROR - INVALID EXPEDITED DATA LENGTH</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=SENDEXPD was issued that specified an expedited data length of 0 or an expedited data length greater than the allowed maximum. The largest expedited data size that can be sent with one macroinstruction invocation is 86 bytes.
<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’002D’</td>
<td>USF6INSC</td>
<td>PARAMETER_ERROR - INVALID SENSE CODE VALUE SPECIFIED</td>
</tr>
</tbody>
</table>

An APPC/MD CONTROL=DEALLOC|DEALLOCQ, QUALIFY=ABNDUSER was specified with a sense code that was not an allocation or abnormal deallocation sense code value.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’002E’</td>
<td>USF6VANV</td>
<td>VECTOR AREA NOT VALID</td>
</tr>
</tbody>
</table>

The application supplied VTAM with a vector area address that is not valid or is write-protected.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’002F’</td>
<td>USF6VALI</td>
<td>VECTOR AREA LENGTH INSUFFICIENT</td>
</tr>
</tbody>
</table>

The application supplied VTAM with a vector area that is smaller than the minimum required size.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0030’</td>
<td>USF6STNV</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
</tbody>
</table>

A storage type indication was not supplied or is not valid. Storage type is required to be specified via the ISTAPC82 mapping DSECT that is mapped within the ISTAPC82 mapping DSECT.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0031’</td>
<td>USF6VALS</td>
<td>PARAMETER_ERROR—SENDRCV_SPECIFIED_WITHOUT_OPTION=BUFFLST</td>
</tr>
</tbody>
</table>

The APPC/MD CONTROL=SENDRCV was issued without specifying a buffer. OPTCD=BUFFLST|XBUFLST is required for this macroinstruction.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0032’</td>
<td>USF6UNXV</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
</tbody>
</table>

An unexpected vector was provided on an APPC/MD request. An input vector is not defined for the APPC/MD.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’002C’</td>
<td>X’0033’</td>
<td>USF6NPV</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_OR_SPECIFIED_INCORRECTLY</td>
</tr>
</tbody>
</table>

A required input vector was either not provided or specified incorrectly on an APPC/MD request.
### Return codes

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0034'</td>
<td>USF6LNSP</td>
<td>PASSWORD_SUBSTITUTION_VALUE_SET_IN_ERROR</td>
</tr>
</tbody>
</table>

The FMH-5 received from the application indicated password substitution in byte 4, bit 3. The session established with the partner does not support password substitution. Reissue the macroinstruction with this bit setting off.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>USF6PENT</td>
<td>PROGRAM ERROR NO TRUNCATION</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 SEND_ERROR verb specifying the TYPE(PROG) parameter; the conversation for the remote program was in a sending state; and the LU 6.2 SEND_ERROR verb did not truncate a logical record. No truncation occurs when a transaction program issues the LU 6.2 SEND_ERROR verb before sending any logical records or after sending a complete logical record. This return code is reported to the local application program when it issues an APPCCMD CONTROL=RECEIVE macroinstruction prior to receiving any logical records or after receiving one or more complete logical records.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>USF6PEPU</td>
<td>PROGRAM ERROR PURGING</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying the TYPE(PROG) parameter, and the conversation for the remote transaction program was in RECEIVE state. The LU 6.2 SEND_ERROR verb might have caused information to be purged. Purging occurs when a transaction program issues the LU 6.2 SEND_ERROR verb in RECEIVE state before receiving all the information sent by the local application, that is, all the information sent prior to the reporting of the PROGRAM_ERROR_PURGING return code to the local application. The purging can occur at the local LU, the remote LU, or both. No purging occurs when a transaction program issues the LU 6.2 SEND_ERROR verb in a CONFIRM state, or in RECEIVE state after receiving all the information sent by the local application. This RCPRI code is normally reported to the local application on an APPCCMD it issues after sending some information to the remote transaction program. However, the RCPRI code can be reported on an APPCCMD the application issues prior to sending any information, depending on the CONTROL and QUALIFY fields of the APPCCMD and when it is issued. The conversation is in RECEIVE state.

**Note:** This code is never reported on an APPCCMD issued on a full-duplex conversation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>USF6PETR</td>
<td>PROGRAM ERROR TRUNCATING</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying the TYPE(PROG) parameter; the conversation for the remote transaction program was in a sending state; and the LU 6.2 SEND_ERROR verb truncated a logical record. Truncation occurs when a transaction program begins sending a logical record and then issues the LU 6.2 SEND_ERROR verb before sending the complete logical record. This return code is reported to the local application on an APPCCMD CONTROL=RECEIVE macroinstruction issued after receiving the truncated logical record. The conversation state is unchanged.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>USF6SENT</td>
<td>SERVICE ERROR NO TRUNCATION</td>
</tr>
</tbody>
</table>
The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying the TYPE(SVC) parameter; the conversation for the remote transaction program was in a sending state; and the LU 6.2 SEND_ERROR verb did not truncate a logical record. No truncation occurs when a transaction program issues the LU 6.2 SEND_ERROR verb before sending any logical records or after sending a complete logical record. This return code is reported to the local application on an APPCCMD CONTROL=RECEIVE macroinstruction it issues prior to receiving any logical records or after receiving one or more complete logical records. The conversation state is unchanged.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>USF6SEPU</td>
<td>SERVICE ERROR PURGING</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying the TYPE(SVC) parameter, and the conversation for the remote transaction program was in RECEIVE state. The LU 6.2 SEND_ERROR verb might have caused information to be purged. Purging occurs when a transaction program issues the LU 6.2 SEND_ERROR verb in RECEIVE state before receiving all the information sent by the local application, that is, all the information sent prior to the reporting of the SERVICE_ERROR_PURGING return code to the local application. The purging can occur at the local LU, the remote LU, or both. No purging occurs when a transaction program issues the LU 6.2 SEND_ERROR verb in a CONFIRM state, or in RECEIVE state after receiving all the information sent by the local application. This return code is normally reported to the local application on an APPCCMD it issues after sending some information to the remote transaction program. However, the return code can be reported on an APPCCMD the application issues prior to sending any information, depending on the CONTROL and QUALIFY fields of the APPCCMD and when it is issued. The conversation is in RECEIVE state.

**Note:** This code is never reported on an APPCCMD issued on a full-duplex conversation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>USF6SETR</td>
<td>SERVICE ERROR TRUNCATING</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying the TYPE(SVC) parameter; the conversation for the remote transaction program was in a sending state; and the LU 6.2 SEND_ERROR verb truncated a logical record. Truncation occurs when a program begins sending a logical record and then issues the LU 6.2 SEND_ERROR verb before sending the complete logical record. This return code is reported to the local application on an APPCCMD CONTROL=RECEIVE macroinstruction issued after receiving the truncated logical record. The conversation state is unchanged.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>USF6RFNR</td>
<td>RESOURCE FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>

A failure occurred that caused the conversation to be prematurely terminated. For example, the session being used for the conversation was deactivated because of a session protocol error. The condition is not temporary, and the application should not try the transaction again until the condition is corrected. The conversation is in END_CONV or FDX_RESET state if no log data is present. If log data is present, the conversation is in PEND_END_CONV_LOG or PEND_RESET_LOG state.

Two common failures are:

- Local LU sends unexpected control information.

  For example, the conversation can be in PENDING_DEALLOCATE state, but something other than a deallocate is received, or an FMH-7 is not received when it is expected.

- Local LU sends unexpected data on the conversation.

  For example, a logical record that is not valid, PS header or FMH-7, might have been received, or a logical record is truncated by something other than an FMH-7.
A failure occurred that caused the conversation to be prematurely terminated. For example, the session being used for the conversation was deactivated because of a session outage, such as a line failure or a modem failure. The application can try the transaction again when the error that caused the session outage has been corrected. The conversation is in END_CONV or FDX_RESET state.

The specified conversation was not in an appropriate state to issue the specified APPCCMD. For example, the application program issued APPCCMD CONTROL=SEND, QUALIFY=DATA, but the conversation was in RECEIVE state. The state of the conversation remains unchanged.

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because the partner LU does not recognize the specified mode name. The local and partner LUs' CNOS parameters are not changed.

The APPCCMD CONTROL=ALLOC, QUALIFY=IMMED macroinstruction issued by the local application program did not execute successfully because there was not a contention-winner session available for use by a new conversation request. This RCPRI code is returned on the unsuccessful APPCCMD.

An FMH-7 was received that contained a sense code not interpreted by VTAM. The unrecognized sense code is passed to the application program through the SENSE field in the RPL extension. The application program must determine whether the sense code is a valid user-supplied sense code or a code that is not valid. The USER_ERROR_CODE_RECEIVED RCPRI code together with the RCSEC subcodes (X'0000' X'0001') form the complete return code that is returned to the application. The subcode specifies whether a negative response preceded the FMH-7 containing the unrecognized sense code. The conversation is in a receiving state.

The FMH-7 containing the unrecognized sense code was received by VTAM after the receipt of a negative response.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>USF6RFRE</td>
<td>RESOURCE FAILURE, RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>USF6STER</td>
<td>STATE ERROR</td>
</tr>
<tr>
<td>X'0054'</td>
<td>X'0000'</td>
<td>USF6URMD</td>
<td>UNRECOGNIZED MODE NAME</td>
</tr>
<tr>
<td>X'0058'</td>
<td>X'0000'</td>
<td>USF6UNSC</td>
<td>UNSUCCESSFUL, SESSION NOT AVAILABLE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>(all)</td>
<td>USF6UECR</td>
<td>USER ERROR CODE RECEIVED</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USF6FNGR</td>
<td>NEGATIVE RESPONSE</td>
</tr>
</tbody>
</table>
The FMH-7 containing the unrecognized sense code was not preceded by a negative response.

The application issued an APPCCMD CONTROL=RCVFMH5, but there is currently no FMH-5 waiting to be received by the application program.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS macroinstruction did not execute successfully because activation for the pending active session failed. For example, the path between the application and the other LU could have been lost.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS macroinstruction did not execute successfully because activating the pending active session would have caused the session limits for the mode name group to be exceeded.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS or QUALIFY=DACTSESS macroinstruction was issued for a session that is no longer pending. The CID for the session is valid but a BIND or CINIT is no longer queued, or the session is being deactivated due to a previous error or request.

VTAM is unable to process the request because of a temporary storage shortage, a resource shortage, or other shortage.

- If a sense code is not provided, a temporary storage shortage has occurred.
- If a sense code is provided indicating insufficient resources, then a storage shortage or other resource shortage has occurred. In either of these cases, the request can be reissued (with EXECRPL, for example.) There is no state change. This return code is reported to the application program to allow time for the problem to diminish or disappear. If VTAM attempts to try the request again, the additional storage might not be available immediately, and the problem might occur again.
- If a sense code is provided other than one for insufficient resources, examine the sense code explanation to determine the action required. In this situation, whether the request can be reissued depends on the information contained in the sense code.
If this return code is received at the completion of an APPCCMD with CONTROL=RECEIVE, OPTCD=(XBUFLST), then a CSM buffer that meets the storage type specified in the XBUFLST-receive vector could not be obtained to receive the data, or other VTAM internal resources required to receive the data could not be obtained. The system is storage constrained. No data is received.

The application can take several possible actions:
- Reissue the APPCCMD several times as a temporary try recovery action again.
- Issue a receive without the XBUFLST specification so the data can be copied into application private storage.
- Explicitly deallocate the conversation via APPCCMD services.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>USF6HALT</td>
<td>HALT ISSUED</td>
</tr>
</tbody>
</table>

The operator has issued a HALT command. Depending on the type of HALT, the application program can no longer issue certain macroinstructions.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>USF6VIYA</td>
<td>VTAM INACTIVE FOR YOUR ACB</td>
</tr>
</tbody>
</table>

The association between VTAM and the application program (ACB) that was established with the OPEN macroinstruction has been broken (the ACB is in the process of being closed). This might have occurred because:
- The application program has elsewhere issued a CLOSE that has not yet completed
- VTAM has become inactive
- A VARY NET,INACT command was issued for the application program.

Any active conversations are placed in END_CONV or FDX_RESET state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>USF6RQAB</td>
<td>REQUEST ABORTED</td>
</tr>
</tbody>
</table>

VTAM has rejected a request because of an error detected while processing the request or because of an error in the associated session, task, or address space. For example, an abend. An abend might or might not be retried.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0080'</td>
<td>X'0000'</td>
<td>USF6DLNR</td>
<td>DEALLOCATE NORMAL</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 DEALLOCATE TYPE(FLUSH) verb. This return code is reported to the application program on an APPCCMD CONTROL=SEND, QUALIFY=ERROR macroinstruction issued when the conversation is in RECEIVE state. The conversation is in END_CONV state. The conversation can be in RECEIVE state or in PEND_RCV_LOG state. This return code applies only to half-duplex conversations.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>USF6STSH</td>
<td>STORAGE SHORTAGE</td>
</tr>
</tbody>
</table>
Indicates VTAM has encountered a storage shortage when attempting to satisfy an APPCCMD CONTROL=RECEIVE or an APPCCMD CONTROL=RCVFMH5, either while storing incoming data or sending a pacing response. There is no state change.

This return code can also be issued when a storage failure occurs while processing an internal DEALLOC FLUSH request. VTAM does internal DEALLOC FLUSH processing when it receives an indication that the partner has issued an abnormal deallocation request on the full-duplex conversation.

The application should issue one of the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions to end the conversation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>USF6CREJ</td>
<td>CANCELED BY REJECT OR ABNORMAL DEALLOCATE</td>
</tr>
</tbody>
</table>

The request, while in progress, was canceled by the issuance of an APPCCMD CONTROL=REJECT or abnormal deallocation APPCCMD, which has requested the termination of the current conversation and, possibly, the session.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>USF6PROE</td>
<td>PARTNER COMMITTED PROTOCOL VIOLATION</td>
</tr>
</tbody>
</table>

The partner LU has violated conversation protocols during the execution of this command. Notification of conversation failure will be received on a subsequent APPCCMD command. There is no state change.

Two common protocol violations are:

- Partner LU sends unexpected control information.
  For example, the conversation can be in PENDING_DEALLOCATE state, but something other than a deallocate is received, or an FMH-7 is not received when it is expected.
- Partner LU sends unexpected data on the conversation.
  For example, a logical record that is not valid, PS header or FMH-7, might have been received, or a logical record is truncated by something other than an FMH-7.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>USF6NOTA</td>
<td>APPLICATION NOT APPC CAPABLE</td>
</tr>
</tbody>
</table>

The application program issued an APPCCMD, but the application program has APPC=NO coded on its APPL definition statement. The APPL definition statement must have APPC=YES coded before the application program can issue APPCCMD macroinstructions.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>USF6SDRJ</td>
<td>INVALID CONDITION FOR SENDING DATA</td>
</tr>
</tbody>
</table>

This indicates that the application program issued an APPCCMD that provided data to be sent an error on a previous QUALIFY=DATAFLU or QUALIFY=DATACON type of send (either CONTROL=SEND, CONTROL=PREPRCV or CONTROL=DEALLOC). However, data remains, held by VTAM, from the error on the previous DATAFLU or DATACON macroinstruction.

Before sending more data, issue a macroinstruction that flushes VTAM’s buffers. An APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction, an APPCCMD CONTROL=SEND, QUALIFY=ERROR macroinstruction, or one of the abnormal termination CONTROL=DEALLOC macroinstructions will flush the send data queue so that processing can continue.
This indicates a temporary storage shortage has occurred while sending data. This RCPRI, RCSEC combination might be returned for one of the macroinstructions:

- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDPROG
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDSERV
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDTIME
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDUSER
- APPCCMD CONTROL=DEALLOC, QUALIFY=DATA
- APPCCMD CONTROL=DEALLOC, QUALIFY=DATAFLU
- APPCCMD CONTROL=PREPRCV, QUALIFY=DATA
- APPCCMD CONTROL=PREPRCV, QUALIFY=DATACON
- APPCCMD CONTROL=SEND, QUALIFY=DATA
- APPCCMD CONTROL=SEND, QUALIFY=DATAFLU
- APPCCMD CONTROL=SEND, QUALIFY=ERROR
- APPCCMD CONTROL=SENDRCV, QUALIFY=DATAFLU.

The current position in the application-supplied data buffer (the area pointed to by the AREA field of the RPL) is returned in RPL6STBF (the current buffer) and RPL6STDS (displacement in the data). All data prior to this buffer or buffer list entry has been sent.

The user has two alternatives when this return code is received.

- Attempt to continue sending data on the conversation by issuing an APPCCMD macroinstruction with the data pointers and length set to reflect the values returned in RPL6STBF and RPL6STDS. The subsequent macroinstruction must be issued with the AREA field set with the RPL6STBF value plus the RPL6STDS value to avoid duplicating any data already sent. The data length (the RECLEN field in the RPL) must also be adjusted to indicate the amount of remaining data. Once the subsequent macroinstruction with the updated data location completes successfully, the conversation can be continued as if the storage shortage did not occur.
- Deactivate the conversation by issuing one of the abnormal termination CONTROL=DEALLOC macroinstructions, or APPCCMD CONTROL=REJECT macroinstructions. Note that REJECT must be issued to deactivate a conversation if the abnormal termination CONTROL=DEALLOC macroinstructions are unsuccessful.

The APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE macroinstruction is issued before the SETLOGON START macroinstruction is issued.
VTAM rejected the APPCCMD because the macroinstruction request conflicts in some way with the capabilities of the session or conversation to which it applies. The REQUEST_NOT_ALLOWED RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the transaction program.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>'X'00A0'</td>
<td>(all)</td>
<td>USF6RNAL</td>
<td>REQUEST NOT ALLOWED</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=SENDEXPD because the negotiated support level of the current session does not support protocols needed to transmit expedited data.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>'X'00A0'</td>
<td>'X'0001'</td>
<td>USF6LNSE</td>
<td>LU PAIR DOES NOT SUPPORT SENDING EXPEDITED DATA</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD because the conversation with which it is associated is in the process of being deallocated or terminated.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>'X'00A0'</td>
<td>'X'0002'</td>
<td>USF6RQBL</td>
<td>REQUEST BLOCKED</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC on a half-duplex conversation because the partner LU is awaiting a change-direction or end-of-chain indicator before sending error information. No expedited information was available to be received.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>'X'00A0'</td>
<td>'X'0003'</td>
<td>USF6RNEX</td>
<td>EXECUTION OF REQUEST TERMINATED</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD because the CONTROL= and QUALIFY= value combination specified is not allowed for a full-duplex conversation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>'X'00A0'</td>
<td>'X'0004'</td>
<td>USF6VNVF</td>
<td>CONTROL/QUALIFY VALUE INVALID FOR FULL-DUPLEX CONVERSATION</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD CONTROL=SENDEXPD,QUALIFY=DATA or an APPCCMD CONTROL=SEND, QUALIFY=RQSEND because the response to a previously issued APPCCMD CONTROL=SENDEXPD,QUALIFY=DATA had not been received from the partner LU.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>'X'00A0'</td>
<td>'X'0005'</td>
<td>USF6EXRO</td>
<td>RSP HAS NOT BEEN RECEIVED FOR A PREVIOUS SENDEXPD REQUEST</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD CONTROL=SENDEXPD,QUALIFY=DATA or an APPCCMD CONTROL=SEND, QUALIFY=RQSEND because the response to a previously issued APPCCMD CONTROL=SENDEXPD,QUALIFY=DATA had not been received from the partner LU.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>'X'00A0'</td>
<td>'X'0006'</td>
<td>USF6NAUT</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
</tbody>
</table>
An application not using VTAM authorized path attempted to use the HPDT interface. The request is disallowed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0008'</td>
<td>USF6ENEL</td>
<td>NAMED RESOURCE NOT ELIGIBLE FOR REQUESTED ALTERATION</td>
</tr>
</tbody>
</table>

A MODIFY DEFINE command with DELETE=UNUSE was issued for an entry in the LU-mode table, but the entry type is not UNUSABLE.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A4'</td>
<td>X'0000'</td>
<td>USF6SPMD</td>
<td>MODE MUST BE RESTORED BEFORE USING</td>
</tr>
</tbody>
</table>

An APPCCMD macroinstruction is issued with a mode name that is pending recovery for persistent LU-LU sessions. Issue the APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE macroinstruction to restore the mode.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A8'</td>
<td>(all)</td>
<td>USF6ENVE</td>
<td>ENVIRONMENT ERROR</td>
</tr>
</tbody>
</table>

A macroinstruction has failed for some reason related to the system environment in which the request was processed. The RCSEC subcode identifies the specific error.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>USF6OSLV</td>
<td>OS LEVEL DOES NOT SUPPORT REQUESTED FUNCTION</td>
</tr>
</tbody>
</table>

A macroinstruction request required the use of an operating system service which is not supported by the active operating system level.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>USF6XMES</td>
<td>SUSPEND FAILURE</td>
</tr>
</tbody>
</table>

VTAM attempted to suspend processing of an APPCCMD macroinstruction issued in either cross-memory mode or in synchronous SRB-mode with OPTCD=KEEPSRB specified. The attempt failed, probably due to conditions in the operation system environment. The application may reissue the request.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>USF6XMER</td>
<td>RESUME FAILURE</td>
</tr>
</tbody>
</table>

VTAM attempted to resume processing of an APPCCMD macroinstruction issued in either cross-memory mode or in synchronous SRB-mode with OPTCD=KEEPSRB specified. The attempt failed. VTAM is unable to post the request complete. If the application has a LOSTERM exit, it will be scheduled with a reason code of 44. For more information about the LOSTERM exit, see z/OS Communications Server: SNA Programming. The RPL is now available for reuse.

Return codes 561
VTAM's processing of an APPCCMD request stored on the SEND queue of a full-duplex conversation was ended because the remote transaction program or LU issued an LU 6.2 architecture verb that canceled further processing of the request. An associated Secondary Return Code value indicates the type of operation that caused the request to be ended.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>(all)</td>
<td>USF6ERIN</td>
<td>ERROR INDICATION RECEIVED</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that the remote transaction program issued a DEALLOCATE verb with TYPE(ABEND_PROG).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>USF6EIAS</td>
<td>DEALLOCATE ABEND PROGRAM</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that the remote transaction program issued a DEALLOCATE verb with TYPE(ABEND_SVC).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>USF6ERAS</td>
<td>DEALLOCATE ABEND SERVICE</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that the remote transaction program issued a DEALLOCATE verb with TYPE(ABEND_TIMER).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>USF6EIAT</td>
<td>DEALLOCATE ABEND TIME</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that an allocation request was rejected by the remote transaction program.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>USF6EIAT</td>
<td>ALLOCATION ERROR</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code other than the Dealocate ABEND, Allocation Error, or Resource Failure codes. The application program must determine whether the sense code is a valid user-supplied sense code or is a code that is not valid.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>USF6EIUN</td>
<td>UNKNOWN ERROR CODE</td>
</tr>
</tbody>
</table>
An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because a failure occurred that caused the conversation to be prematurely terminated. The application can try the transaction again when the error that caused the session outage has been corrected.

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because a failure occurred that caused the conversation to be prematurely terminated. The condition is not temporary, and the application should not try the transaction again until the condition is corrected.

VTAM rejected an APPCCMD because there was an inappropriate name translation. The NAME_RESOLUTION_ERROR RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the transaction program.

VTAM rejected an APPCCMD because the LUNAME specified on the macroinstruction was found in a VARIANT_NAME entry in the LU-mode table.

VTAM rejected an APPCCMD because the BIND RSP contained an LUNAME that is different from the associated name in the SUPPLIED_NAME entry in the LU-mode table. The association of names for the partner LU had previously occurred.

VTAM rejected an APPCCMD because the LUNAME returned in the BIND RSP was found in a VARIANT_NAME entry in the LU-mode table. The association of names for the partner LU has not occurred.

VTAM rejected an APPCCMD because the LUNAME returned in the BIND RSP was found in a SUPPLIED_NAME entry in the LU-mode table.
VTAM rejected an APPCCMD because the LUNAME contained in the BIND RSP was found in a SUPPLIED_NAME entry in the LU-mode table. The SUPPLIED_NAME entry was different than the entry used in the session initiation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>USF6NRNM</td>
<td>PARTNER NETWORK NAME MISMATCH</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD because the NETID contained in the BIND RSP was different than that previously saved in the LU-mode table for that LUNAME.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>USF6NRAV</td>
<td>LUNAME FOUND IN AN UNUSABLE_NAME ENTRY</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD because the LUNAME specified on the macroinstruction was found in an UNUSABLE_NAME entry in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>USF6NRRE</td>
<td>NAME RETURNED FOUND IN AN UNUSABLE_NAME ENTRY</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD because the partner LU returned an LUNAME in the BIND response that was found in an UNUSABLE_NAME entry in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>USF6NRDN</td>
<td>LU NAME FOUND IN A DISASSOCIATED_NAME ENTRY</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD macroinstruction request or an operator command because the LU name specified is a DISASSOCIATED_NAME entry. This type of entry has no mode values and thus has no sessions. The LU name was previously a VARIANT_NAME entry but is no longer associated with a SUPPLIED_NAME entry.

If the request or operator command was to display information about the LU, reissue the request with LOGMODE=0 and any LU-specific information will be returned.

If the request was for an allocate, a CNOS must be issued to establish mode information before the allocate can be retried.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>(all)</td>
<td>USF6CSME</td>
<td>CSM_DETECTED_ERROR</td>
</tr>
</tbody>
</table>

CSM detected an error. The CSM_DETECTED_ERROR RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the transaction program.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>USF6NSPC</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>

CSM detected a problem during APPCCMD processing of the request. The specific reason for the error is not passed back to the APPCCMD application.
Upon receipt of this return code the application can:

- Optionally consider the error temporary and try the request again several times.

  Note that it is possible that the error may not recur. This temporary error condition could occur in the case where a VTAM-built parameter list to CSM is randomly corrupted on a particular request, but not on a subsequent request.

- Consider the error permanent and terminate the conversation.

Refer to z/OS Communications Server: CSM Guide for more information about these CSM errors.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>X'0002'</td>
<td>USF6IBTK</td>
<td>CSM_DETECTED_ERROR—INVALID_BUFFER_TOKEN_SPECIFIED</td>
</tr>
</tbody>
</table>

The communications storage manager (CSM) detected a problem during APPCCMD processing of the request. The specific reason for the error is that CSM detected that the CSM buffer token being used for the APPCCMD is not a valid CSM buffer token.

Upon receipt of this return code the application can:

- Check the current buffer pointer (RPL6STBF) in the RPL extension to determine the address of the buffer list entry that was processed when the error occurred.

- Optionally consider the error temporary and try the request again several times.

  Note that it is possible that the error may not recur. This temporary error condition could occur in the case where a VTAM-built parameter list to CSM is randomly corrupted on a particular request, but not on a subsequent request.

- Consider the error permanent and terminate the conversation.

- Continue using the conversation with a different CSM buffer.

Refer to z/OS Communications Server: CSM Guide for more information about these CSM errors.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>X'0003'</td>
<td>USF6IIID</td>
<td>CSM_DETECTED_ERROR—INVALID_INSTANCE_ID_SPECIFIED</td>
</tr>
</tbody>
</table>

The communications storage manager (CSM) detected a problem during APPCCMD processing of the request. The specific reason for the error is that CSM detected that the instance ID portion of the CSM buffer token being used for the APPCCMD is not a valid CSM instance ID. Because the instance ID is not valid, it is possible that the CSM buffer being specified on the APPCCMD has been previously freed and a new instance ID has been assigned to the storage by CSM.

Upon receipt of this return code the application can:

- Check the current buffer pointer (RPL6STBF) in the RPL extension to determine the address of the buffer list entry that was processed when the error occurred.

- Optionally consider the error temporary and try the request again several times.

  Note that it is possible that the error may not recur. This temporary error condition could occur in the case where a VTAM-built parameter list to CSM is randomly corrupted on a particular request, but not on a subsequent request.

- Consider the error permanent and terminate the conversation.

- Continue using the conversation with a different CSM buffer.

Refer to z/OS Communications Server: CSM Guide for more information about these CSM errors.
RTNCD, FDB2 information for LU 6.2

While most of the LU 6.2 feedback information from errors is found in the RCPRI and RCSEC fields, some error return codes in the RPL RTNCD and FDB2 fields are meaningful for LU 6.2 applications. The X'00', X'0B' combination in the RPL indicates some problem might have occurred while the macroinstruction was executing. RCPRI and RCSEC should be used for further diagnosis. The other RTNCD, FDB2 combinations refer to attempts to start an LU 6.2 session independent of VTAM or attempts to use non-APPCCMD macroinstructions for APPCCMD functions. The following information shows the relevant codes.

<table>
<thead>
<tr>
<th>RTNCD</th>
<th>FDB2</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>X'0B'</td>
<td>USF6APPC</td>
<td>CONDITIONAL COMPLETION FOR APPCCMD</td>
</tr>
</tbody>
</table>

Some type of error might have occurred on an APPCCMD macroinstruction. For further problem determination, refer to the primary and secondary return codes in the RPL extension. These fields are RPL6RCPR and RPL6RCSC.

<table>
<thead>
<tr>
<th>RTNCD</th>
<th>FDB2</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'04'</td>
<td>X'05'</td>
<td>USFNQN</td>
<td>SYMBOLIC NAME KNOWN BY NETWORK-QUALIFIED NAME ONLY</td>
</tr>
</tbody>
</table>

A real-to-symbolic translation request is made, and NIBNET is filled in with a network identifier, but VTAM cannot provide a symbolic name. VTAM knows this resource only by its network-qualified name. No symbolic name represents this resource. Do one of the following actions:

- Use the network-qualified name
- Define a symbolic name to represent this resource

<table>
<thead>
<tr>
<th>RTNCD</th>
<th>FDB2</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'10'</td>
<td>X'13'</td>
<td>USF6APRJ</td>
<td>ATTEMPT TO START 6.2 SESSION: REQUEST REJECTED</td>
</tr>
</tbody>
</table>

An LU 6.2 application program has tried to start an LU 6.2 session independent of VTAM. No pending sessions have been disturbed. This occurs when an OPNDST is issued with an LU 6.2 user-specified BIND.

<table>
<thead>
<tr>
<th>RTNCD</th>
<th>FDB2</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'10'</td>
<td>X'14'</td>
<td>USF6APST</td>
<td>ATTEMPT TO START 6.2 SESSION: PENDING SESSION TERMINATED</td>
</tr>
</tbody>
</table>

An LU 6.2 application program has tried to start an LU 6.2 session independent of VTAM. The pending session has been terminated. This occurs when the LOGMODE specified on an OPNDST resolves to an LU 6.2 BIND or when OPNSEC is issued for an LU 6.2 BIND.

<table>
<thead>
<tr>
<th>RTNCD</th>
<th>FDB2</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'10'</td>
<td>X'15'</td>
<td>USF6APIS</td>
<td>AN APPCCMD MUST BE ISSUED</td>
</tr>
</tbody>
</table>

An OPNDST or CLSDST has been issued for a pending LU 6.2 session. An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS or QUALIFY=DACTSESS macroinstruction must be issued for this session.
An application program issued a non-APPCCMD macroinstruction to establish an LU 6.2 session, or issued a non-APPCCMD macroinstruction against a current LU 6.2 session.
Chapter 3. DSECTs

This chapter contains the LU 6.2 DSECTs. For general information on the use and purpose of DSECTs, refer to z/OS Communications Server: SNA Programming.

The DSECTs are shown as an abbreviated form of an assembler listing. The first column indicates the offsets within the DSECT and is the “LOC” column of an assembler listing. The object code, address columns and statement number columns of the listing, however, are not included. The source statements and comments are found next to the offset column. All numbers in the offset column are in hexadecimal.

### BIND image (ISTDBIND)

<table>
<thead>
<tr>
<th>LOC</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>ISTDBIND DSECT</td>
</tr>
<tr>
<td>000000</td>
<td>BINfmty DS C</td>
</tr>
<tr>
<td></td>
<td>BINfmt EQU X’F0’</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINfmt (FORMAT)</td>
</tr>
<tr>
<td></td>
<td>BINfmt0 EQU X’00’</td>
</tr>
<tr>
<td></td>
<td>BINtype EQU X’0F’</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINtype (TYPE)</td>
</tr>
<tr>
<td></td>
<td>BINneg0 EQU X’00’</td>
</tr>
<tr>
<td></td>
<td>BINneg0 EQU X’01’</td>
</tr>
<tr>
<td></td>
<td>BINcold EQU X’01’</td>
</tr>
<tr>
<td>000001</td>
<td>BINfm DS C</td>
</tr>
<tr>
<td></td>
<td>* PROFILE</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINfm - FUNCTION</td>
</tr>
<tr>
<td></td>
<td>MANAGEMENT PROFILE</td>
</tr>
<tr>
<td></td>
<td>BINfm19 EQU X’13’</td>
</tr>
<tr>
<td>000002</td>
<td>BINTS DS C</td>
</tr>
<tr>
<td></td>
<td>* PROFILE</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINTS (TRANSMISSION</td>
</tr>
<tr>
<td></td>
<td>SERVICES PROFILE)</td>
</tr>
<tr>
<td></td>
<td>BINT07 EQU X’07’</td>
</tr>
<tr>
<td></td>
<td>BINT04 EQU X’04’</td>
</tr>
<tr>
<td></td>
<td>BINT03 EQU X’03’</td>
</tr>
<tr>
<td></td>
<td>BINT02 EQU X’02’</td>
</tr>
<tr>
<td></td>
<td>BINT01 EQU X’01’</td>
</tr>
<tr>
<td></td>
<td>BINT00 EQU X’00’</td>
</tr>
<tr>
<td>000003</td>
<td>BNPrip DS C</td>
</tr>
<tr>
<td></td>
<td>* sending fm data</td>
</tr>
<tr>
<td></td>
<td>BNPchn EQU X’80’</td>
</tr>
<tr>
<td></td>
<td>0 = SINGLE RU CHAINS</td>
</tr>
<tr>
<td></td>
<td>BNPmch EQU X’40’</td>
</tr>
<tr>
<td></td>
<td>0 = SINGLE OUTSTANDING</td>
</tr>
<tr>
<td></td>
<td>* CHAIN (IMMEDIATE)</td>
</tr>
<tr>
<td></td>
<td>* REQUEST MODE</td>
</tr>
<tr>
<td>BNPchnr EQU X’30’</td>
<td>CHAIN RESPONSE PROTOCOL(SEE BINSchnr BELOW FOR VALUES)</td>
</tr>
<tr>
<td>BINrsv01 EQU X’OC’</td>
<td>RESERVED</td>
</tr>
<tr>
<td>BNPcmp EQU X’02’</td>
<td>1 = COMPRESSION MAY BE USED</td>
</tr>
<tr>
<td>0 = COMPRESSION MUST NOT BE USED</td>
<td></td>
</tr>
<tr>
<td>BNPseb EQU X’01’</td>
<td>1 = PRIMARY MAY SEND EB</td>
</tr>
<tr>
<td>0 = PRIMARY WILL NOT SEND EB</td>
<td></td>
</tr>
<tr>
<td>000004</td>
<td>BINsecp DS C</td>
</tr>
<tr>
<td></td>
<td>* sending fm data</td>
</tr>
<tr>
<td></td>
<td>BINSchn EQU X’80’</td>
</tr>
<tr>
<td></td>
<td>0 = SINGLE RU CHAIN</td>
</tr>
<tr>
<td>BINSchnr EQU X’30’</td>
<td>CHAIN RESPONSE PROTOCOLS (TYPE OF RESPONSES ASKED FOR BY REQUESTS FROM PRIMARY/SECONDARY)</td>
</tr>
</tbody>
</table>

© Copyright IBM Corp. 2000, 2019
BINNYRSP EQU X'30' DEFINITE OR EXCEPTION RESPONSE
BINDFRSP EQU X'20' DEFINITE RESPONSE
BINEXRSP EQU X'10' EXCEPTION RESPONSE
BINNORSP EQU X'00' NO RESPONSE
BINSV02 EQU X'0C' RESERVED
BINCMP EQU X'02' 1 = COMPRESSION MAY BE USED
* 0 = COMPRESSION MUST NOT BE USED
BINSSEB EQU X'01' 1 = SECONDARY MAY SEND EB
* 0 = SECONDARY WILL NOT SEND EB
000005 BINCMP DS C COMMON LU PROTOCOLS
BINWBREQ EQU X'80' WHOLE-BINS-REQUIRED INDICATOR
* BINFMHD EQU X'40' 1 = FM HEADERS MAY BE USED
* 0 = FM HEADERS MUST NOT BE USED
BINBRAK EQU X'20' 1 = BRACKETS WILL BE USED AND RESET STATE IS BETWEEN-BRACKETS
* 0 = BRACKETS WILL NOT BE USED OR, IF USED, RESET STATE IS IN-BRACKETS
BINKTR EQU X'10' 1 = CONDITIONAL BRACKETS TERMINATION
* 0 = UNCONDITIONAL BRACKETS TERMINATION
BINALT EQU X'08' 1 = ALTERNATE CODE MAY BE USED
* 0 = ALTERNATE CODE MUST NOT BE USED
BINSV04 EQU X'06' RESERVED
BINQUE EQU X'01' BIND-QUEUEING INDICATOR
000006 BINCMP2 DS C COMMON LU PROTOCOLS
BINFMTRM EQU X'C0' SEND/RECEIVE MODE
* VALUES FOR BINFMTRM
BINMSTSL EQU X'C0' RESERVED
BINHDXXF EQU X'80' HDX FULL DIRECTION
BINHDXC EQU X'40' HDX CONTENTION
BINFLDPX EQU X'00' FULL DUPLEX
BINRCVR EQU X'20' 1 = SYMMETRIC RESPONSIBILITY FOR RECOVERY
* 0 = CONTENTION LOSER (SEE BINDKFS BELOW) RESPONSIBLE FOR RECOVERY
BINBKFS EQU X'10' 1 = PRIMARY IS BRACKETS WINNER; SECONDARY IS BRACKETS BIDDER AND CONTENTION LOSER
* 0 = SECONDARY IS BRACKETS WINNER; PRIMARY IS BRACKETS BIDDER AND CONTENTION LOSER
BINA ASCC EQU X'0C' ALTERNATE CODE PROCESSING IDENTIFIER
* 00=ASCII7
* 01=ASCII8
BINCNTLV EQU X'02' CONTROL VECTORS ARE INCLUDED
* AFTER THE SLU NAME
BINCONR EQU X'01' SET RESET STATE FOR HDX FLIP-FLOP (E.G. AT START OF SESSION)
* 1 = PRIMARY SENDS FIRST WHEN DATA TRAFFIC
* 0 = SECONDARY SENDS FIRST
000007 BINTSU DS CL6 TS USAGE
00000D BINPRSVC DS CL12 PRESENTATION SERVICES
000019 BINCRCTL DS CL1 CRYPTOGRAPHY CONTROL BYTE
* VALUES FOR BINCRCTL
BINNOCRY EQU X'00' NO CRYPTOGRAPHY
BINCRCYCA EQU X'09' CAPABLE OF CRYPTOGRAPHY
BINCRCYSL EQU X'19' SELECTIVE CRYPTOGRAPHY
BINCRCYR EQU X'39' REQUIRED CRYPTOGRAPHY
* BINCEUMP EQU X'C0' EU/PRIVATE CRYPTOGRAPHY FLAGS
* VALUES FOR BINCEUMP
**BINCEUPS** EQU X'80'  SYSTEM KEY, PRIVATE PROTOCOL
**BINCEUPP** EQU X'40'  PRIVATE KEY, PRIVATE PROTOCOL
**BINCEUNP** EQU X'00'  NO PRIVATE/EU PROTOCOL

**BINCSESS** EQU X'30'  SESSION LEVEL CRYPTOGRAPHY FLAGS

**BINCSENP** EQU X'00'  NO CRYPTOGRAPHY
**BINCSESP** EQU X'10'  SELECTIVE CRYPTOGRAPHY
**BINCSESR** EQU X'30'  REQUIRED CRYPTOGRAPHY

**BINCELEN** EQU X'0F'  LENGTH OF CRYPTOGRAPHY FIELD

**BINCLEN** EQU X'0F'  LENGTH OF CRYPTOGRAPHY FIELD

**00001A** BINPRIML DS C  PRIMARY LU NAME LENGTH
**00001B** BINPRIMN DS CL8  PRIMARY LU NAME

* INCLUDE FOR COMPATIBILITY *

**000023** ORG BINPRIMN
**00001B** BINPRIM DS 8C  PRIMARY LU NAME

**000023** BINUSEL DS C  USER DATA LENGTH
**BINUSE** EQU *  USER DATA
**BINUSERD** EQU X'00'  USER DATA LENGTH DEFAULT

* OVERLAY FOR 'BINTSU' (TS USAGE)

**000024** ORG BINTSU

**000067** BINAPACE DS C  SLU SEND PACING
**BINS2ST** EQU X'80'  NUMBER OF PACING STAGES FROM
**SLU TO PLU** (NOTE-REVERSE OF
**BINPS2ST))

**BINRSV07** EQU X'40'  RESERVED
**BINAPCM** EQU X'3F'  SLU SEND PACING COUNT

**000008** BINRPACE DS C  SLU RECEIVE PACING
**BINASPI** EQU X'80'  ADAPTIVE SESSION PACING INDICATOR

**BINRSV07** EQU X'40'  RESERVED
**BINRPACE** DS 1C2  RU SIZES
**BINRUSZ** DS C  SLU MAXIMUM SEND RU SIZE
**BINRUSZ** EQU X'80'  RU SIZE IS SPECIFIED

**00000A** BINPRUSZ DS C  PLU MAXIMUM SEND RU SIZE
**BINPRSS** EQU X'80'  RU SIZE IS SPECIFIED

* VALUES FOR BINRUSZ AND BINRUSZ (RU SIZES) EXCEPT RU SIZE

**00000B** BINRUSZ DS C  PLU SEND PACING
**BINPS2ST** EQU X'80'  NUMBER OF PACING STAGES FROM
**PLU TO SLU** (NOTE-REVERSE OF
**BINPS2ST))

**BINRSV07** EQU X'40'  RESERVED
**BINRUSZ** DS C  PLU RECEIVE PACING

**00000C** BINBPACE DS C  PLU RECEIVE PACING
**BINRSV44** EQU X'40'  RESERVED
**BINRUSZ** DS C  PLU RECEIVE PACING

**00000D** ORG BINPRSVC

**00000D** BINLUP DS C  PS PROFILE

* OVERLAY FOR 'BINPRSVC' (PRESENTATION SERVICES)

*********************************************************************
*
**00000D** ORG BINPRSVC

**00000E** BINPSCHR DS CL11  PS PROFILE DEPENDENT

* OVERLAY FOR 'BINPRSVC' (PRESENTATION SERVICES)

*********************************************************************

DSECTs 571
* **************************************************
* OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
* FOR PS PROFILE 1)
* **************************************************
000019 ORG BINPSCHR
00000E BINLUP1 DS X PS PROFILE 1 FMHS AND DSP
BINFMHS1 EQU X'F0' FM HEADER SUBSET
* VALUES FOR BINFMHS1
BINFMSC1 EQU X'30' DATA MANAGEMENT SUBSET
BINFMSC2 EQU X'20' TYPE 1 HEADERS
BINFMSC1 EQU X'10' TYPE 1 HEADERS WITH
* RESTRICTIONS
BINFMSC0 EQU X'00' NO FM HEADERS ALLOWED
BINDSP1 EQU X'00' DATA STREAM PROFILE
* VALUES FOR BINDSP1 (DATA STREAM PROFILE)
BINDSP1C EQU X'01' BASIC CONTROLS, CARDS MAY
* SPAN RUS
BINDSP0C EQU X'00' BASIC CONTROLS
00000F BINPLUS1 DS 0XL5 PLU USAGE
00000F BINPFMF1 DS 0XL2 FMH SUBSET DEPENDENT
* FLAGS
00000F BINPFMB1 DS X FIRST BYTE
000010 BINPFMB2 DS X SECOND BYTE
000011 BINPDSP1 DS 0XL2 DATA STREAM FLAGS FOR
* DSP0 AND DSP1
000011 BINPDSP1 DS X FIRST BYTE
000012 BINPDSP2 DS X SECOND BYTE
000013 BINPFMD1 DS X MEDIA FLAGS
000014 BINSLU1 DS 0XL5 SLU USAGE
000014 BINSFMF1 DS 0XL2 FMH SUBSET DEPENDENT
* FLAGS
000014 BINSFMF1 DS 0XL2 FMH SUBSET DEPENDENT
000014 BINPFMB1 DS X FIRST BYTE
000015 BINPFMB2 DS X SECOND BYTE
000016 BINPDSP1 DS 0XL2 DATA STREAM FLAGS FOR
* DSP0 AND DSP1
000016 BINSDB1 DS X FIRST BYTE
000017 BINSDB2 DS X SECOND BYTE (RESERVED)
000018 BINSMD1 DS X MEDIA FLAGS
* * FLAGS FOR LU PROFILE 1
* * FLAGS FOR BINPFMB1 AND BINSFMF1 (FIRST BYTE OF FM
* * HEADER FLAGS)
* BINDESTS EQU X'80' 0 = TWO DESTINATIONS MAY
* BE OUTSTANDING
* 1 = THREE DESTINATIONS MAY
* BE OUTSTANDING
* BINCMCPT EQU X'40' 0 = WILL NOT SEND COMPACATION
* TABLE/WILL NOT BE QUERIED
* FOR COMPACATION TABLES
* 1 = MAY SEND COMPACATION
* TABLE/MAY BE QUERIED FOR
* COMPACATION TABLES
* BINPDIF EQU X'20' 0 = PDIF WILL NOT BE SENT
* 1 = PDIF MAY BE SENT
* BINRSV09 EQU X'1F' RESERVED FOR FMHS1
* ADDITIONAL FLAGS FOR FMHS3
* BINADD6 EQU X'19' 0 = KEYED DIRECT DATA SET
* WILL NOT BE SENT
* 1 = KEYED DIRECT DATA SET
* MAY BE SENT
* BINSDESI EQU X'08' 0 = SEQUENTIAL DATA SETS
* WILL NOT BE SENT
* 1 = SEQUENTIAL DATA SETS
* MAY BE SENT
* BINSAI EQU X'04' 0 = SEQUENTIAL ACCESS TO
* ADDRESSED DIRECT DATA
* SET WILL NOT BE SENT
* 1 = SEQUENTIAL ACCESS TO
* ADDRESSED DIRECT DATA
* SET MAY BE SENT
* BINSID EQU X'02' 0 = SERIES ID NOT
* SUPPORTED (WITH STATUS
* IN REPLY)
* 1 = SERIES ID SUPPORTED
* (WITH STATUS IN REPLY)
* BINARR EQU X'01' 0 = ADD REPLICATE,
* REPLACE REPLICATE NOT
* SUPPORTED
* 1 = ADD REPLICATE,
* REPLACE REPLICATE
* SUPPORTED
* FLAGS FOR BINPFMB2 AND BINFMB2 (SECOND BYTE OF FM HEADER FLAGS)
BINRSV17 EQU X'FF'
* RESERVED FOR FMHS1
* ADDITIONAL FLAGS FOR FMHS3
BINRSV16 EQU X'80'
* 0 = QUERY FOR DESTINATION
* SELECTION NOT SUPPORTED
* 1 = QUERY FOR DESTINATION
* SELECTION SUPPORTED
* BINCSDS EQU X'20'
* 0 = CREATE / SCRATCH /
* SCRATCH ALL DATA
* 1 = CREATE / SCRATCH /
* SCRATCH ALL DATA
* SET NOT ALLOWED
* BINXFPD EQU X'10'
* 0 = EXECUTE PROGRAM OFFLINE
* NOT ALLOWED
* 1 = EXECUTE PROGRAM OFFLINE
* ALLOWED
BINRSV11 EQU X'0F'
* RESERVED FOR FMHS3
* FLAGS FOR 'BINPDSB1 AND BINSDB1' (PLU/SLU DATA STREAM
* FLAGS FOR DSP0 AND DSP1)
* NL AND FF MAY BE SENT IN ANY SUBSET. EACH SUBSET BELOW CONTAINS
* EVERY PRECEDING SUBSET (E.G. IF AN LU CAN SEND THE HORIZONTAL
* FORMAT SUBSET, IT CAN ALSO SEND THE FULL BASE SET)
BININTR EQU X'80'
* 0 = FULL BASE SET DATA
* STREAM (BS,CR,LF,ENP,
* INP,HT,VT) WILL NOT
* BE SENT
* 1 = FULL BASE SET DATA
* STREAM (BS,CR,LF,ENP,
* INP,HT,VT) MAY BE
* SENT
* BINHFDS EQU X'40'
* 0 = HORIZONTAL FORMAT,
* DATA STREAM(SHF) WILL
* NOT BE SENT
* 1 = HORIZONTAL FORMAT,
* DATA STREAM(SHF) MAY
* BE SENT
* BINVTDS EQU X'20'
* 0 = VERTICAL FORMAT
* DATA STREAM (SVF)
* WILL NOT BE SENT
* 1 = VERTICAL FORMAT
* DATA STREAM (SVF)
* MAY BE SENT
* BINVSDS EQU X'10'
* 0 = VERTICAL CHANNEL DATA
* STREAM (SVF(CHANNELS),SCF,
* SEL) WILL NOT BE
* SENT
* 1 = VERTICAL CHANNEL DATA
* STREAM (SVF(CHANNELS),SCF,
* SEL) MAY BE SENT
* BINSLD EQU X'08'
* 0 = SLD WILL NOT BE SENT
* 1 = SLD MAY BE SENT
* RESERVED
* BINRSV40 EQU X'06'
* BINDOCMT EQU X'80'
* 0 = DOCUMENT FORMAT WILL
* NOT BE SENT
* 1 = DOCUMENT FORMAT
* FLAGS FOR BINPDSB2
* BINUAINT EQU X'80'
* 0 = SLU WILL INITIATE
* ATTENDED
* 1 = SLU WILL INITIATE
* UNATTENDED
* BINUAALT EQU X'40'
* 0 = DURING SESSION SLU
* WILL NOT ALTERNATE
* BETWEEN ATTENDED AND
* UNATTENDED
* 1 = DURING SESSION SLU
* WILL ALTERNATE
* BETWEEN ATTENDED AND
* UNATTENDED
* BINRSV41 EQU X'3F'
* RESERVED
* FLAGS FOR BINPMED1 AND BINSMED1 (PLU/SLU MEDIA FLAGS)
BINDOCMT EQU X'80'
* 0 = DOCUMENT FORMAT WILL
* NOT BE SENT
* 1 = DOCUMENT FORMAT
* DSECTs 573
BINCARD EQU X'40'  
*      MAY BE SENT
* 0 = CARD FORMAT WILL NOT BE SENT
* 1 = CARD FORMAT MAY BE SENT
BINXCHNG EQU X'29'  
*      EXCHANGE MEDIA FORMAT
* 0 = WILL NOT BE SENT
* 1 = MAY BE SENT
BINDISK EQU X'10'  
*      DISK FORMAT WILL NOT BE SENT
* 0 = BE SENT
* 1 = DISK FORMAT MAY BE SENT
BINXCDF EQU X'08'  
*      EXTENDED CARD FORMAT
* 0 = WILL NOT BE SENT
* 1 = MAY BE SENT
BINXDOCF EQU X'04'  
*      EXTENDED DOCUMENT
* 0 = FORMAT WILL NOT BE SENT
* 1 = EXTENDED DOCUMENT MAY BE SENT
BINCDEDS EQU X'02'  
*      SLU MAY SEND CD EVERY EDS
* 0 = NO SLU MAY SEND CD EVERY EDS
* 1 = SLU MUST SEND CD EVERY EDS
* (THIS FLAG APPLIES TO BINPMED1)
BIN1CMP1 EQU X'02'  
*      APPLIES ONLY to BINSMED1
* (SEE BINCMP1 AND BINCMP2)
BIN1CMP2 EQU X'01'  
* (SEE BINCMP1 AND BINCMP2)
*********************************************************************
* OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
* FOR PS PROFILE 2)
*********************************************************************
000019 ORG BINPSCHR
00000E BINFLAG DS XL1  
* DEVICE FLAG
00000F BINSDF5 DS XL4  
* EXTENDED 3270 DATA STREAM
000013 BINSSCRZ DS 0XL5  
* PRESENTATION SPACE SIZE
000013 BINSPRIR DS FL1  
* PRIMARY (DEFAULT) NUMBER OF ROWS
000014 BINSALTR DS FL1  
* ALTERNATE NUMBER OF ROWS
000015 BINSALTC DS FL1  
* ALTERNATE NUMBER OF COLUMNS
000017 BINSIDFRS DS FL1  
* PRESENTATION SPACE SIZE
* VALUES FOR BINPRESZ (PRESENTATION SPACE SIZE)
BINPSZRC EQU X'7F'  
* DEFAULT AND ALTERNATE
* SIZES AS DEFINED IN
* DEFAULT, ALTERNATE
* ROW/COL FIELDS
* PRESENTATION SPACE HAS
* FIXED SIZE AS DEFINED IN
* DEFAULT ROW/COL FIELDS
* DEFAULT ROW/COL VALUES IN
* DEFAULT ROW/COL FIELDS
* 24X80 DEFAULT UNDEFINED ALTERNATE
* DO WRITE STRUCTURED FIELD QUERY
* TO IDENTIFY ALTERNATE
* 24X80 ROW X COLUMN
* 12X40 ROW X COLUMN
* UNDEFINED ROW X COLUMN
* TO IDENTIFY ALTERNATE
* PRIMARY (DEFAULT) NUMBER OF ROWS
* PRIMARY (DEFAULT) NUMBER OF COLUMNS
* SLU MAY SEND CD EVERY EDS
* ALTERNATE NUMBER OF ROWS
* ALTERNATE NUMBER OF COLUMNS
* PRESENTATION SPACE
*********************************************************************
* OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
* FOR PS PROFILE 3)
*********************************************************************
000019 ORG BINPSCHR
00000E BINSDF5 DS XL4  
* RESERVED
000013 BINSFRDS DS 0XL4  
* PRESENTATION SPACE SIZE
000013 BINSFRDR DS FL1  
* PRIMARY (DEFAULT) NUMBER OF ROWS
000014 BINSFRDC DS FL1  
* PRIMARY (DEFAULT) NUMBER OF COLUMNS
000015 BINFLRFS DS FL1  
* ALTERNATE NUMBER OF ROWS
000016 BINFRFAC DS FL1  
* ALTERNATE NUMBER OF COLUMNS
000017 BINBDESC DS FL1  
* PRESENTATION SPACE
* SIZE SPECIFICATION:
  * 0 = MAXIMUM
  * 1 = 480 CHAR
  * 2 = 1920 CHAR
  * X'7E' = FIXED SIZE (SEE BINFRDR AND BINFRDC)
  * X'7F' = VARIABLE SIZE AS DEFINED BY BINFRSZ

BINBFSIZ EQU X'7F'
BINBFSZF EQU X'7E'

000018 BIN3CMP DS X
BIN3CMP1 EQU X'02'
BIN3CMP2 EQU X'01'

*********************************************************************
OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
FOR PS PROFILE 4)
*********************************************************************

000019 ORG BINPSCHR

00000E BINPSNDO DS 0XL4
PLU SEND CAPABILITY

00000E BINPDSPP DS X
PRINTER DATA STREAM

 * PROFILE
  * BINPBSDP EQU X'80'
  * BASE DATA STREAM PROFILE
  * 0 = NOT SUPPORTED
  * 1 = SUPPORTED

BINRSV46 EQU X'40'
RESERVED

BINPJOB EQU X'20'
JOB SCS SUBSET

 * 0 = NOT SUPPORTED
  * 1 = SUPPORTED

BINRSV47 EQU X'10'
RESERVED
BINWPRAW EQU X'08'
WORD PROCESSING RAW FORM

 * 0 = NOT SUPPORTED
  * 1 = SUPPORTED

BINRSV48 EQU X'07'
RESERVED

00000F BINADSPP DS X
ADDITIONAL DATA STREAM

 * PROFILE
  * BINRSV49 EQU X'80'
  * BASE DATA STREAM PROFILE
  * 0 = CARD NOT SUPPORTED
  * 1 = CARD SUPPORTED

BINADCD EQU X'40'
RESERVED

000010 BINCBDSP EQU X'80'
BASE DATA STREAM PROFILE

 * 0 = NOT SUPPORTED
  * 1 = SUPPORTED

BINCJOB EQU X'20'
JOB SCS SUBSET

 * 0 = NOT SUPPORTED
  * 1 = SUPPORTED

BINCJOB EQU X'20'
JOB SCS SUBSET

 * 0 = NOT SUPPORTED
  * 1 = SUPPORTED

000011 BINFNUP DS X
FM/FMH USAGE

BINSSDAT EQU X'80'
RESERVED
BINSDS0 EQU X'60'
00= 1 LEVEL DESTINATION

 * SELECTION SUSPENSION
  * 01= 2 LEVEL DESTINATION
  * SELECTION SUSPENSION
  * STACK
  * 10= RESERVED
  * 11= 3 LEVEL DESTINATION
  * SELECTION SUSPENSION
  * STACK

BINRSV52 EQU X'1E'
RESERVED
BINKIXS EQU X'01'
0 = SLU NEED NOT RECEIVE CD ON EVERY EDS

 * 1 = SLU MUST RECEIVE CD ON EVERY EDS

000012 BINSNDO DS 0XL4
SLU SEND CAPABILITY

000012 BINFDSPP DS X
PRINTER DATA STREAM

 * PROFILE (SEE BINPDSPP)

000013 BINADPSPP DS X
ADDITIONAL DATA STREAM

 * PROFILE (SEE BINADPSPP)

000014 BINCBLSP DS X
CONSOLE (SEE BINCBLSP)

000015 BINFNHSU DS X
FM/FMH USAGE (SEE BINFNHSU)

 * BINFNHSU: MEANING FOR BINKIXS IS: 0 = PLU NEED NOT RECEIVE CD ON EVERY EDS, 1 = PLU MUST RECEIVE CD ON EVERY EDS

000016 BINC0 DS X
CODE SELECTION

DSECTs 575
BINCSOR EQU X'F0' REPERTOIRE
BINCSOE EQU X'80' EBCDIC
BINCSOAI EQU X'40' ASCII / ISCII / ITA#5
BINRSV30 EQU X'20' RESERVED
BINRSV31 EQU X'10' RESERVED
BINCS0C1 EQU X'0C' 00 = CODE 0 (MAIN CODE)
* SELECTION IS EBCDIC
* 01 = CODE 0 (MAIN CODE)
* SELECTION IS ASCII / ISCII / ITA#5
BINCSOC2 EQU X'03' 00 = CODE 1 (ALTERNATE
* CODE SELECTION IS EBCDIC
* 01 = CODE 1 (ALTERNATE CODE SELECTION IS ASCII / ISCII / ITA#5
* 000017 BINGENCO DS X GENERAL CHARACTERISTICS
* BINGENCO DS X GENERAL CHARACTERISTICS
BINRSV32 EQU X'C0' RESERVED
BINRSV33 EQU X'10' RESERVED
BINIAO EQU X'08' 0 = PLU MAY SEND DATA FIRST
* 1 = SLU MUST SEND DATA FIRST
* BINIAO EQU X'08' 0 = PLU WILL INITIATE ATTENDED
* 1 = SLU WILL INITIATE ATTENDED
* BINAAO EQU X'04' 0 = SLU WILL NOT ALTERNATE BETWEEN ATTENDED
* 1 = SLU MAY ALTERNATE BETWEEN ATTENDED AND UNATTENDED
* BINRSV34 EQU X'03' 000018 BINRSV35 DS X RESERVED
* 000018 BINRSV35 DS X RESERVED
**********************************************************************
* OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
* FOR PS PROFILE 6)
*********************************************************************
000019 ORG BINPSCHR
00000E BINLULEV DS X LU-6 LEVEL
BINL02 EQU X'02' LEVEL 2
00000F BINRSV36 DS XL6 RESERVED
000015 BINFLG0 DS X FLAGS
BINDSSSP EQU X'80' DISTRIBUTED SYSTEMS SECURITY
* @=EXTENDED SECURITY MECHANISMS
* ARE NOT SUPPORTED
* 1=AT LEAST ONE SECURITY MECHANISM IS SUPPORTED
* BINDESS EQU X'40' Extended Security Sense Codes
* @= Extended security sense codes will not be accepted on incoming FMH-7s
* 1= Extended security sense codes will be accepted on incoming FMH-7s
* 000016 BINFLG1 DS X LU 6.2 FLAGS
BINCLS5 EQU X'10' ACCESS SECURITY SUBFIELD SUPPORT:
* @= ACCESS SECURITY INFORMATION FIELD WILL NOT BE ACCEPTED ON INCOMING FMH-5S
* 1= ACCESS SECURITY INFORMATION FIELD WILL BE ACCEPTED ON INCOMING FMH-5S
* BINSAPS EQU X'08' SESSION LEVEL SECURITY PROTOCOL
* * BINDPWS EQU X'04' Password Substitution Support:
* @= Substituted passwords will not be accepted on incoming FMH-5s
* 1= Substituted passwords will be accepted on incoming FMH-5s
* BINAVFS EQU X'02' ALREADY - VERIFIED FUNCTION SUPPORT

FMH-5 (ISTFM5)

LOC      SOURCE STATEMENT
000000  ISTFM5   DSECT                FMH5 MAPPING
000000  FM5BASE  DS    0CL10              FIXED LENGTH BASE
000000  FSMLENTH DS    X                  LENGTH FIELD
000001  FM5FLAG1 DS    X                  FLAG FIELDS 1
000001  FM5CONCT EQU   X'80'                CONCATENATION INDICATOR
000001  FM5TYP   EQU   X'7F'                FMH TYPE MASK
000001  FM5TYPE5 EQU   X'05'                IBM ARCHITECTED FMH5
FM5TYPE DS XL2 FMH5 TYPE
FM5ATTCH EQU X'02FF' FMH5 TYPE = ATTACH
FM5FLAG2 DS X FLAG BYTE
FM5UIDAV EQU X'08' USER ID ALREADY VERIFIED
FM5PV1 EQU X'40' USER ID SIGNED ON
FM5PV2 EQU X'20' USER ID SIGN ON
FM5PWS EQU X'10' PASSWORD SUBSTITUTION
  * IF THIS BIT IS 0 AND A PASSWORD
  * IS PRESENT IT IS IN THE CLEAR
  * IF THIS BIT IS 1 AND A PASSWORD
  * IS PRESENT IT IS A SUBSTITUTED
  * PASSWORD
FM5PPIP EQU X'08' PIP PRESENT AFTER FMH5
FM5DSSPR EQU X'04' DISTRIBUTED SYSTEMS SECURITY
  * PRESENT AFTER FMH5 (AND PIP
  * GDS IF PRESENT). IF THIS BIT
  * IS ON, FM5UIDAV, FM5PV1, AND
  * FM5PV2 MUST BE ZERO AS WELL AS
  * THE SECURITY ACCESS SUBFIELDS.
FM5LNFLP DS X LENGTH OF FIXED LENGTH PARAMETERS
FM5FXLEN DS 0XL3 FIXED LENGTH PARAMETERS
FM5RSCTP DS X RESOURCE TYPE
FM5BASIC EQU X'D0' BASIC CONVERSATION
FM5MAPED EQU X'D1' MAPPED CONVERSATION
FM5FDBAS EQU X'D2' FULL-DUPLEX BASIC CONVERSATION
FM5FDMAP EQU X'D3' FULL-DUPLEX MAPPED
  * CONVERSATION
FM5CNFLP DS X LENGTH OF TRANSACTION PROGRAM NAME
  (NOT INCLUDING THIS BYTE)
FM5TPNAM DS 0X TRANSACTION PROGRAM NAME
FM5ASI DSECT ACCESS SECURITY INFORMATION
  SUBFIELDS
FM5LNASI DS X LENGTH OF ASI SUBFIELDS
  (NOT INCLUDING THIS BYTE)
FM5ASEC DS 0X CONTAINS ALL ACCESS SECURITY
  SUBFIELDS. THESE SUBFIELDS ARE
  * MAPPED BY THE FM5ACCSE DSECT.
  * THERE MAY BE ZERO OR MORE OF
  * THESE SUBFIELDS, AND EACH MUST
  * BE SEPARATELY MAPPED BY THE
  * FM5ACCSE DSECT.
FM5LUOW1 DSECT LOGICAL UNIT OF WORK IDENTIFIER
  FIELD
FM5LNLW DS X LENGTH OF LUOW ID
  (NOT INCLUDING THIS BYTE)
FM5LUWI DS 0X LUOW ID
FM5LNFQN DS X LENGTH OF FULLY QUALIFIED LU NAME
  (NOT INCLUDING THIS BYTE)
FM5FQNAM DS 0X FULLY QUALIFIED LU NAME
FM5LUOW2 DSECT LUOW
FM5LUWIN DS XL6 LUOW INSTANCE NUMBER
FM5LUWSN DS XL2 LUOW SEQUENCE NUMBER
FM5CVCOR DSECT CONVERSATION CORRELATOR
FM5LNCSCS DS X LENGTH OF CONVERSATION CORRELATOR
  OF SENDER
  (NOT INCLUDING THIS BYTE)
FM5CCS DS 0X CONVERSATION CORRELATOR OF SENDING
  TRANSACTION
FM5SEQNM DSECT SEQUENCE NUMBER MAP
FM5LNSNMD DS X LENGTH OF SEQUENCE NUMBER
  (NOT INCLUDING THIS BYTE)
FM5SNM DS XL8 SEQUENCE NUMBER

**********************************************************************
ACCESS SECURITY SUBFIELD
  * THIS DSECT IS USED TO MAP EACH ACCESS SECURITY SUBFIELD. THESE
  * SUBFIELDS ARE ALL CONTAINED IN THE FIELD 'FM5ASEC'. YOU MUST

578 z/OS Communications Server: SNA Programmer's LU 6.2 Reference
DETERMINE HOW MANY SUBFIELDS ARE SPECIFIED, AND DETERMINE THE LENGTH OF EACH OF THE SUBFIELDS.

*****************************************************************************
000000 FM5ACCSE DSECT ACCESS SECURITY SUBFIELD
000000 FM5ASLL DS X SUBFIELD LENGTH
* (NOT INCLUDING THIS BYTE)
000001 FM5ASTY DS X SUBFIELD TYPE
FM5ASIPR EQU X'00' PROFILE
FM5ASIPW EQU X'01' PASSWORD
FM5ASIID EQU X'02' USER ID
000002 FM5ASDA DS 0X SUBFIELD DATA

*****************************************************************************
000000 FM5PIPFM DSECT PIP FORMAT
000000 FM5PIPLN DS XL2 PIP LENGTH (INCLUDING THIS BYTE)
000002 FM5PIPSGD DS XL2 GDS INDICATOR
FM5PIPF5 EQU X'12F5' PIP VARIABLE
000004 FM5PIPSF DS 0X ZERO OR MORE PIP SUBFIELDS, EACH OF WHICH HAS THE FOLLOWING FORMAT
000000 FM5PIPSM DSECT PIP SUBFIELD MAP
000000 FM5PIPSL DS XL2 SUBFIELD LENGTH (INCLUDING THIS BYTE)
000002 FM5PIPSG DS XL2 GDS INDICATOR
FM5PIPE2 EQU X'12E2' PIP SUBFIELD
000004 FM5PIPSD DS 0X SUBFIELD DATA

*****************************************************************************
000000 ISTRPL6X DSECT
000000 RPL6AREA DS 0CL112 START OF APPC EXTENSION
000000 RPL6CID DS CL4 CONTROL BLOCK IDENTIFIER
000004 RPL6REQ DS XL1 TYPE OF APPCCMD
000005 RPL6QUAL DS XL1 SUBTYPE OF APPCCMD
000006 DS XL2 RESERVED
000008 RPL6CNVD DS XL4 CONVERSATION ID
000010 RPL6USR DS XL4 USER FIELD
000012 RPL6SSID DS XL8 SESSION IDENTIFICATION
000028 RPL6RC DS 0XL4 RPL6 RETURN CODE
000028 RPL6RCPR DS XL2 PRIMARY RETURN CODE
00002A RPL6CFTX DS XL2 CONXMOD INDICATORS
00002B RPL6TYPE EQU X'0C' TYPE INDICATOR
00002C RPL6FILL EQU X'80' FILL INDICATOR
RPL6GFTX EQU X'08' CONFTXT INDICATOR
00002D RPL6NAMU EQU X'03' NAME USE REQUESTED WHEN APPLICATION IS ACTING AS A GENERIC RESOURCE
00002E RPL6FLG1 DS XL1 FIRST INDICATORS BYTE
RPL6FLG2 DS XL1 SECOND INDICATORS BYTE
RPL6FLG3 DS XL1 THIRD INDICATORS BYTE
RPL6FLG1 EQU X'80' RESERVED
RPL6FILL EQU X'80' FILL INDICATOR
RPL6CFTX EQU X'08' CONFTXT INDICATOR
RPL6NAMU EQU X'03' NAME USE REQUESTED WHEN APPLICATION IS ACTING AS A GENERIC RESOURCE
RPL6LOCK EQU X'80' LOCKS INDICATOR
RPL6DECREQ EQU X'60' DEACTIVATION REASON CODE
RPL6EXDR EQU X'10' EXPEDITED DATA RECEIVED

*****************************************************************************

RPL extension (ISTRPL6X)
**RPL6CMOD EQU** X'0C'               **CONMODE INDICATOR**
**RPL6LAST EQU** X'03'               **LAST INDICATOR**
**00002F RPL6FLG4 DS** XL1               **FOURTH INDICATORS BYTE**
**RPL6AFFN EQU** X'C0'               **GENERIC RESOURCE AFFINITY OWNER**
**RPL60EQ EQU** X'3F'               **RESERVED 3F**
**000030 RPL6LU DS** CL8               **NAME OF LU**
**000033 RPL6MODE DS** CL8               **MODE NAME**
**000040 RPL6WHAT DS** 0XL2               **WHAT RECEIVED INDICATOR**
**000040 RPL6RCV1 DS** XL1               **WHAT RECEIVED INDICATOR**
**RPL6WDAT EQU** X'80'               **WHATRCV=DATA**
**RPL6WDAC EQU** X'40'               **WHATRCV=DATA_COMPLETE**
**RPL6WDAI EQU** X'20'               **WHATRCV=DATA_INCOMPLETE**
**RPL6WSND EQU** X'10'               **WHATRCV=SEND**
**RPL6WCFM EQU** X'08'               **WHATRCV=CONFIRM**
**RPL6WDLOG EQU** X'04'               **WHATRCV=DEALLOCATE**
**RPL6WPSH EQU** X'02'               **WHATRCV=LOG_DATA**
**RPL6WPSH EQU** X'01'               **WHATRCV=PS_HEADER**
**000041 RPL6RCV2 DS** XL1               **RESERVED FOR BIT MASK FOR THE**
**RPL6WPSI EQU** X'80'               **WHATRCV=PARTIAL_PS_HEADER**
**RPL6RTUN DS** XL1               **RETURNED INDICATORS AS**
**RPL6RMH5 EQU** X'80'               **FMH5RCV INDICATOR**
**RPL6RLOG EQU** X'40'               **LOGRCV INDICATOR**
**RPL6RSIG EQU** X'20'               **SIGRCV INDICATOR**
**RPL6CLSA EQU** X'10'               **PARTNER LU ACCEPTS SECURITY**
**RPL6AVFA EQU** X'08'               **PARTNER LU ACCEPTS REQUESTS FOR**
**RPL6PV EQU** X'04'               **PARTNER LU ACCEPTS REQUESTS FOR**
**RPL6CRYP EQU** X'03'               **ENCRYPTION LEVEL**
**000042 RPL6MH5L DS** XL1               **LENGTH OF THE FMH 5 RECEIVED**
**000043 RPL6CCST DS** XL1               **CURRENT CONVERSATION STATE**
**000045 RPL6ACTV DS** XL1               **RPL6 ACTIVE INDICATOR**
**RPL6DETP DS** XL1               **DEACTIVATION TYPE CODE**
**RPL6EXDL DS** XL1               **LENGTH OF EXPEDITED DATA**
**000048 RPL6TID DS** A               **TASK ID**
**RPL6MID DS** A               **MACHINE ID**
**RPL6TIX DS** A               **TASK INDEX OF CURRENTLY**
**RPL6VAOL DS** A               **VTAM-TO-APPL VECTORLIST AREA LENGTH**
**RPL6VAIA DS** A               **VTAM-to-APPL VECTORLIST AREA**
**RPL6END EQU** *               **END OF RPL6**

***********************************************************************
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6AFFN.**
**THEY REPRESENT THE "LUAFFIN=" VALUE.**
***********************************************************************
**RPL6NAFF EQU** X'00'               **LUAFFIN NOT SPECIFIED**
**RPL6NAAF EQU** X'80'               **LUAFFIN=NOTAPPL**
**RPL6AAFF EQU** X'00'               **LUAFFIN=APPL**

***********************************************************************
**THE FOLLOWING CONSTANT VALUES ARE THOSE SPECIFIED IN THE**
**EXPEDITED DATA FLOW CONTROL RU "SIGNAL".**
***********************************************************************
**RPL6SIGI EQU** X'00010001'               **SIGNAL DATA RETURNED TO APPLICATION**

***********************************************************************
**THE FOLLOWING CONSTANT IS DEFINED AS A SYMBOLIC REFERENCE**
**TO THE APPC CONTROL BLOCK ID (RPL6).**
***********************************************************************
**RPL6ID EQU** 'APPC'               **APPC CONTROL BLOCK ID**

***********************************************************************
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6REQ.**
***********************************************************************
* THEY REPRESENT THE "CONTROL=" VALUE. *

***********************************************************************
RPL6ALLC EQU X'10' ALLOC
RPL6PLOC EQU X'11' PREALLOC
RPL6SFMG EQU X'12' SENDFMH5
RPL6SRV EQU X'20' RESETRCV
RPL6DEAL EQU X'30' DEALLOC
RPL6DEAQ EQU X'31' DEALLOCQ
RPL6OPR EQU X'40' OPRCNTRL
RPL6PRC EQU X'50' PREPRCV
RPL6RFH5 EQU X'60' RCVFMH5
RPL6RCV EQU X'70' RECEIVE
RPL6RCVX EQU X'71' RCVEXPD
RPL6RJCT EQU X'80' SEND
RPL6SEND EQU X'90' SEND
RPL6SDX EQU X'91' SENDEXPD
RPL6SETS EQU X'A0' SETSEXD
RPL6STS EQU X'B0' TESTSTA
***********************************************************************
* THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6QUAL. *
* THEY REPRESENT THE "QUALIFY=" VALUE. *
* ***********************************************************************
RPL6NQUA EQU X'00' NULL QUALIFER
RPL6APRG EQU X'01' ABNDPROG
RPL6ASRV EQU X'02' ABNDSERV
RPL6ATIM EQU X'03' ABNDTIME
RPL6AUSR EQU X'04' ABNDUSER
RPL6ANY EQU X'05' ANY
RPL6CNOS EQU X'06' CNOS
RPL6CFRM EQU X'07' CONFIRM
RPL6CFMD EQU X'08' CONFMRMD
RPL6DATA EQU X'09' DATA
RPL6DCON EQU X'0A' DACACON
RPL6DFLU EQU X'0B' DATAFLU
RPL6DFIN EQU X'0C' DEFINE
RPL6DSPY EQU X'0D' DISPLAY
RPL6ERR EQU X'0E' ERROR
RPL6FLSH EQU X'0F' FLUSH
RPL6RQSD EQU X'10' RQSEND
RPL6SPEC EQU X'11' SPEC
RPL6ACT EQU X'12' ACTSEXD
RPL6DACT EQU X'13' DACTSEXD
RPL6ALCD EQU X'14' ALLOCD
RPL6IMED EQU X'15' IMME
RPL6CWIN EQU X'16' CONWIN
RPL6SESN EQU X'17' SESSION
RPL6CONV EQU X'18' CONV
RPL6SUSP EQU X'19' SUSPEND
RPL6RESM EQU X'1A' RESUME
RPL6REST EQU X'1B' RESTORE
RPL6SYNB EQU X'1C' SYNCBEG
RPL6SYNE EQU X'1D' SYNCEND
RPL6CNP EQU X'1E' CONVGRP
RPL6SESF EQU X'1F' WHENFREE
RPL6IANY EQU X'20' IANY
RPL6ISP EQU X'21' ISPEC
RPL6QALL EQU X'22' ALL
RPL6IALL EQU X'23' IALL
***********************************************************************
* THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6FILL. *
* THEY REPRESENT THE "FILL=" VALUE. *
***********************************************************************
RPL6BUFF EQU X'00' BUFF
RPL6LL EQU X'80' LL
***********************************************************************
* THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6CD *
* THEY REPRESENT THE "CD=" VALUE *
***********************************************************************
RPL6CDIM EQU X'00' "CD=IMMED"
RPL6CDEF EQU X'40' "CD=DEFER"
***********************************************************************
* THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6CFTX *
* THEY REPRESENT THE "CONFTXT=" VALUE. *
*
RPL6LINO EQU X'00'  NO INFORMATION RETURNED
RPL6LINS EQU X'02'  LU NAME, MODE NAME, AND LM TABLE INFORMATION RETURNED
RPL6LIAL EQU X'04'  ALL INFORMATION IN RESTORE STRUCTURE RETURNED

***********************************************************************

RPL6RSET EQU X'00'  RESET (INITIAL STATE)

** HALF-DUPLEX CONVERSATION STATES **
RPL6SND EQU X'01'  SEND
RPL6RECV EQU X'02'  RECEIVE
RPL6RVCF EQU X'03'  RECEIVE CONFIRM
RPL6RVCSD EQU X'04'  RECEIVE CONFIRM SEND
RPL6PNDD EQU X'06'  PEND DEALLOCATE
RPL6PNDS EQU X'09'  PENDING SEND
RPL6PRVL EQU X'0A'  PENDING RCV LOG

** FULL-DUPLEX CONVERSATION STATES **
RPL6FDRS EQU X'80'  FDX RESET (FINAL)
RPL6FDSD EQU X'81'  FDX SEND/RECEIVE
RPL6FDS O EQU X'82'  FDX SEND-ONLY
RPL6FDSR EQU X'83'  FDX RECEIVE-ONLY
RPL6FRVL EQU X'84'  FDX PENDING SEND/RCV LOG
RPL6FRRS EQU X'85'  FDX PENDING RCV-ONLY LOG
RPL6FRSL EQU X'86'  FDX PENDING RESET LOG

** PENDING CONVERSATION ALLOCATION **
RPL6PALC EQU X'FF'  PENDING ALLOCATE

***********************************************************************

RPL6TCLP EQU X'0F'  CLEANUP
RPL6TPVL EQU X'FE'  PROTOCOL VIOLATION

***********************************************************************

RPL6CNON EQU X'00'  NONE
RPL6CSLE EQU X'01'  SELECTIVE DATA ENCRYPTION
RPL6CMAN EQU X'03'  MANDATORY DATA ENCRYPTION

DSECTs 583
Figure 2. Layout of the RPL extension (part 1 of 3)
<table>
<thead>
<tr>
<th>Flag byte</th>
<th>Primary return code</th>
<th>Secondary return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPL6FLG1</td>
<td>RPL6RCPR (RCPRI)</td>
<td>RPL6RCSC (RCSEC)</td>
</tr>
<tr>
<td>RPL6FLG2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPL6FLG3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPL6FLG4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Flag byte:
- RPL6FLG1: (FILL, CD SLS, CONFTXT, LIST)
- RPL6FLG2: (RTSRTRN, CONXMOD, TYPE, NAMEUSE)
- RPL6FLG3: (LOCKS, DERC, EXDR, CONMODE, LAST)
- RPL6FLG4: (LUAFFIN)

Primary return code - RPL6RCPR (RCPRI)
Secondary return code - RPL6RCSC (RCSEC)

Current conversation state - RPL6CCST (CONSTATE)
RPL in use - RPL6ACTV
Session deactivation type code - RPL6DECT (DEACTYP)
Expedited type code - RPL6EXDL

Figure 3. Layout of the RPL extension (part 2 of 3)
CNOS session limits data structure (ISTSLCNS)

<table>
<thead>
<tr>
<th>LOC</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>ISTSLCNS DSECT</td>
</tr>
<tr>
<td>000000</td>
<td>SLCSESSL DS H</td>
</tr>
<tr>
<td>*</td>
<td></td>
</tr>
<tr>
<td>000004</td>
<td>SLCMCWP DS H</td>
</tr>
<tr>
<td>000006</td>
<td>SLCPARMS DS XL1</td>
</tr>
<tr>
<td></td>
<td>SLCRAL EQU X'80'</td>
</tr>
<tr>
<td></td>
<td>SLCRAP EQU X'40'</td>
</tr>
<tr>
<td></td>
<td>SLCPRSL EQU X'20'</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SLCALL EQU X'10'</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SLCSSLU EQU X'08'</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SLCLNONE EQU X'00'</td>
</tr>
<tr>
<td></td>
<td>SLCLCONV EQU X'04'</td>
</tr>
<tr>
<td></td>
<td>SLCLA VFA EQU X'02'</td>
</tr>
<tr>
<td></td>
<td>SLCLPV EQU X'01'</td>
</tr>
</tbody>
</table>

Figure 4. Layout of the RPL extension (part 3 of 3)
### DEFINE/DISPLAY session limits data structure (ISTSLD)

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLDLUPAR</td>
<td>DS 0XL40</td>
<td>LU SPECIFIC FIELDS</td>
</tr>
<tr>
<td>SLDLUI</td>
<td>DS XL1</td>
<td>LU SPECIFIC FIELDS - BYTE 1</td>
</tr>
<tr>
<td>SLDCAP</td>
<td>EQU X'C0'</td>
<td>LU'S SESSION CAPABILITY MASK</td>
</tr>
<tr>
<td>SLPARR</td>
<td>EQU X'C0'</td>
<td>PARALLEL SESSION CAPABLE</td>
</tr>
<tr>
<td>SLDPNDGP</td>
<td>EQU X'80'</td>
<td>PENDING PARALLEL STATE</td>
</tr>
<tr>
<td>SLDPNDGS</td>
<td>EQU X'40'</td>
<td>PENDING SINGLE STATE</td>
</tr>
<tr>
<td>SLDSSINGL</td>
<td>EQU X'00'</td>
<td>SINGLE SESSION CAPABLE</td>
</tr>
<tr>
<td>SLDSCSBK</td>
<td>EQU X'20'</td>
<td>CONFIRM, SYNC POINT, BACKOUT SUPPORTED</td>
</tr>
<tr>
<td>SLDCONF</td>
<td>EQU X'10'</td>
<td>CONFIRM SUPPORTED</td>
</tr>
<tr>
<td>SLDSYNRT</td>
<td>EQU X'00'</td>
<td>SYNCHRONIZATION LEVEL NOT SET</td>
</tr>
<tr>
<td>SLDCNVCP</td>
<td>EQU X'C0'</td>
<td>CONVERSATION CAPABILITY FIELD</td>
</tr>
<tr>
<td>SLPCLSA</td>
<td>EQU X'40'</td>
<td>SUBFIELDS ON FMH5</td>
</tr>
<tr>
<td>SLPAVFA</td>
<td>EQU X'20'</td>
<td>PARTNER LU ACCEPTS REQUESTS FOR ALREADY VERIFIED FUNCTION</td>
</tr>
<tr>
<td>SLPBV</td>
<td>EQU X'02'</td>
<td>LOCAL LU ACCEPTS REQUESTS FOR PERSISTENT VERIFICATION</td>
</tr>
<tr>
<td>SLDLVFA</td>
<td>EQU X'04'</td>
<td>LOCAL LU ACCEPTS REQUESTS FOR ALREADY VERIFIED FUNCTION</td>
</tr>
<tr>
<td>SLDPPV</td>
<td>EQU X'10'</td>
<td>PARTNER LU ACCEPTS REQUESTS FOR PERSISTENT VERIFICATION</td>
</tr>
<tr>
<td>SLDLCLA</td>
<td>EQU X'08'</td>
<td>LOCAL LU ACCEPTS SECURITY SUBFIELDS ON BMH5 FROM THIS</td>
</tr>
<tr>
<td>SLDLVFA</td>
<td>EQU X'02'</td>
<td>PARTNER LU ACCEPTS REQUESTS FOR PERSISTENT VERIFICATION</td>
</tr>
<tr>
<td>SLDLUN</td>
<td>EQU X'01'</td>
<td>PARTNER LU NAME</td>
</tr>
<tr>
<td>SLDL014</td>
<td>DS XL1</td>
<td>LENGTH OF FULLY QUALIFIED</td>
</tr>
<tr>
<td>SLDL015</td>
<td>DS XL1</td>
<td>APPL NETWORK NAME BEING USED</td>
</tr>
<tr>
<td>SLDL016</td>
<td>DS XL1</td>
<td>GENERIC NAME IS BEING USED</td>
</tr>
<tr>
<td>SLDL017</td>
<td>DS XL1</td>
<td>FULLY QUALIFIED PARTNER LU NAME</td>
</tr>
<tr>
<td>SLDL018</td>
<td>DS XL1</td>
<td>NAME - USE SLDL01N FOR</td>
</tr>
<tr>
<td>SLDL019</td>
<td>DS XL1</td>
<td>ACTUAL LENGTH OF NAME</td>
</tr>
<tr>
<td>SLDL020</td>
<td>DS X'C0'</td>
<td>CONVERSATION CAPABILITY FIELD</td>
</tr>
<tr>
<td>SLDL021</td>
<td>DS X'80'</td>
<td>FULL-DUPLEX OR HALF-DUPLEX, EXPEDITED DATA ALLOWED</td>
</tr>
<tr>
<td>SLDL022</td>
<td>DS X'40'</td>
<td>HALF-DUPLEX CONVERSATIONS ONLY</td>
</tr>
<tr>
<td>SLDL023</td>
<td>DS X'00'</td>
<td>CAPABILITY IS UNKNOWN</td>
</tr>
<tr>
<td>SLDL024</td>
<td>DS X'20'</td>
<td>RESERVED</td>
</tr>
<tr>
<td>SLDL025</td>
<td>DS X'01'</td>
<td>APPLICATION NAME USE FIELD</td>
</tr>
<tr>
<td>SLDL026</td>
<td>DS X'02'</td>
<td>INDICATES FORM OF LU NAME USED BY PARTNER LU</td>
</tr>
<tr>
<td>SLDL027</td>
<td>DS X'03'</td>
<td>PARTNER LU</td>
</tr>
<tr>
<td>SLDL028</td>
<td>DS X'00'</td>
<td>NAME USE NOT YET KNOWN</td>
</tr>
<tr>
<td>SLDL029</td>
<td>DS X'01'</td>
<td>USERVAR NAME IS BEING USED</td>
</tr>
<tr>
<td>SLDL030</td>
<td>DS X'02'</td>
<td>APPL NETWORK NAME IS BEING USED</td>
</tr>
<tr>
<td>SLDL031</td>
<td>DS X'03'</td>
<td>GENERIC NAME IS BEING USED</td>
</tr>
</tbody>
</table>

**LOC**: SOURCE STATEMENT

000000  ISTSLD  DSECT SESSION LIMITS - DEFINE/DISPLAY

* BEGINING OF LU SPECIFIC FIELDS

000000  SLDLUPAR DS 0XL40

000000  SLDLUI   DS XL1

000000  SLDCAP   EQU X'C0'

000000  SLPARR   EQU X'C0'

000000  SLDPNDGP EQU X'80'

000000  SLDPNDGS EQU X'40'

000000  SLDSSINGL EQU X'00'

000000  SLDSCSBK EQU X'20'

000000  SLDCONF  EQU X'10'

000000  SLDSYNRT EQU X'00'

000000  SLDCNVCP EQU X'C0'

000000  SLPCLSA  EQU X'40'

000000  SLPAVFA  EQU X'20'

000000  SLPBV   EQU X'02'

000000  SLDLVFA  EQU X'04'

000000  SLDLCLA  EQU X'08'

000000  SLDLVFA  EQU X'02'

000000  SLDLVFA  EQU X'04'

000000  SLDLUN   EQU X'01'

000000  SLDL014  DS XL1

000000  SLDL015  DS XL1

000000  SLDL016  DS X'C0'

000000  SLDL017  DS X'80'

000000  SLDL018  DS X'40'

000000  SLDL019  DS X'00'

000000  SLDL020  DS X'20'

000000  SLDL021  DS X'01'

000000  SLDL022  DS X'02'

000000  SLDL023  DS X'03'

000000  SLDL024  DS X'20' - X'01'

000000  SLDL025  DS X'01'

000000  SLDL026  DS X'02'

000000  SLDL027  DS X'03'

000000  SLDL028  DS X'00'

000000  SLDL029  DS X'01'

000000  SLDL030  DS X'02'

000000  SLDL031  DS X'03'

DSECTs 587
Raw text:

<table>
<thead>
<tr>
<th>SLDSUPNM</th>
<th>EQU   X'00'</th>
<th>SUPPLIED_NAME ENTRY</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLDCVNM</td>
<td>EQU   X'01'</td>
<td>RCVD_NAME ENTRY</td>
</tr>
<tr>
<td>SLDVARNM</td>
<td>EQU   X'02'</td>
<td>VARIANT_NAME ENTRY</td>
</tr>
<tr>
<td>SLDUNUNM</td>
<td>EQU   X'03'</td>
<td>UNUSABLE_NAME ENTRY</td>
</tr>
<tr>
<td>SLDDISNM</td>
<td>EQU   X'04'</td>
<td>DISASSOC_NAME ENTRY</td>
</tr>
</tbody>
</table>

Reserved

END OF LU SPECIFIC FIELDS

BEGINING OF MODE SPECIFIC FIELDS

SLDDSESL DS   H  DEFINED SESSION LIMIT
SLDDMCWL DS   H  DEFINED NUMBER OF CONTENTION WINNER SESSIONS -- LOCAL LU
SLDDMCWP DS   H  DEFINED NUMBER OF CONTENTION WINNER SESSIONS -- PARTNER LU
SLDEFPA DS   XL1  DEFINED PARAMETERS
SLDDRSPLO EQU X'00'  DEFINED ACCEPTANCE OF DEACTIVATION RESPONSIBILITY, IF SET THEN THE LOCAL LU WILL ACCEPT RESPONSIBILITY
SLDDRALE EQU X'40'  DEFINED ACCEPTANCE OF REQUEST TO DRAIN QUEUED ALLOCS, IF SET THEN THE LOCAL LU WILL ACCEPT THE REQUEST
SLDELEMT EQU X'20'  MODE DELETION INDICATOR, IF SET APPL WILL ALLOW DELETION OF MODE
SLDAUTOS EQU X'10'  AUTOSES SPECIFIED AS ON DEFINE
SLDMDSUS EQU X'08'  MODE PENDING RECOVERY
SLDDSALE EQU X'04' - X'01'  RESERVED

SLDCNSPA DS   XL1  CNOS PARAMETERS
SLDDRATL EQU X'80'  DRAINING OF LOCAL LU
SLDDRATL EQU X'40'  DRAINING OF PARTNER LU

SESSION LIMIT
SLDMCWLS DS   H  MINIMUM NUMBER OF CONTENTION WINNER SESSIONS -- LOCAL LU
SLDMCWPL DS   H  MINIMUM NUMBER OF CONTENTION WINNER SESSIONS -- PARTNER LU
SLDSTAUS DS   H  AUTO ACTIVATE LIMIT
SLDSESSC DS   H  CURRENT SESSION COUNT
SLDNICLDS DS   H  NUMBER OF CURRENT CONTENTION WINNER SESSIONS -- LOCAL LU
SLDNICLPS DS   H  NUMBER OF CURRENT CONTENTION WINNER SESSIONS -- PARTNER LU
SLDFREEE DS   H  NUMBER OF FREE SESSIONS
SLDQALLC DS   H  NUMBER OF ALLOCATE REQUEST WAITING FOR FREE SESSIONS

RESERVED

END OF MODE SPECIFIC FIELDS

END OF ISTSLD

### Restore data structure (ISTSREST)

<table>
<thead>
<tr>
<th>LOC</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>ISTSREST DSECT</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>000000</td>
<td>SRENAME DS CL8</td>
</tr>
<tr>
<td>000008</td>
<td>SREMODE DS CL8</td>
</tr>
<tr>
<td>000010</td>
<td>SREXTAD DS A</td>
</tr>
<tr>
<td>000014</td>
<td>SRESLAD DS A</td>
</tr>
<tr>
<td>000018</td>
<td>SRESESAD DS A</td>
</tr>
<tr>
<td>00001C</td>
<td>SREMFLGS DS XL2</td>
</tr>
<tr>
<td>00001E</td>
<td>SREMSRS EQU X'80'</td>
</tr>
<tr>
<td>000020</td>
<td>SRESESCT DS H</td>
</tr>
<tr>
<td>000028</td>
<td>SRENETID DS CL8</td>
</tr>
<tr>
<td>000028</td>
<td>SREEND DS 0X</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>000000</td>
<td>SRESESS DSECT</td>
</tr>
<tr>
<td>000000</td>
<td>SRESNXA DS A</td>
</tr>
<tr>
<td>000004</td>
<td>SRESFLGS DS XL3</td>
</tr>
<tr>
<td>00001E</td>
<td>SREPConv EQU X'80'</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>000000</td>
<td>SRESPND EQU X'40'</td>
</tr>
</tbody>
</table>

---

588  z/OS Communications Server: SNA Programmer's LU 6.2 Reference
**Status data structure (ISTSTATD)**

<table>
<thead>
<tr>
<th>LOC</th>
<th>SOURCE STATEMENT</th>
<th>DSECT</th>
<th>TESTSTAT INFORMATION ENTRY</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>ISTSTATD DSECT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000000</td>
<td>STATENTL DS XL2</td>
<td></td>
<td>LENGTH OF THIS ENTRY</td>
</tr>
<tr>
<td>000002</td>
<td>STATENTT DS X</td>
<td></td>
<td>ENTRY TYPE</td>
</tr>
<tr>
<td></td>
<td>STATNRM EQU X'01'</td>
<td></td>
<td>NORMAL DATA INFORMATION ENTRY</td>
</tr>
<tr>
<td></td>
<td>STATXPD EQU X'02'</td>
<td></td>
<td>EXPEDITED DATA INFORMATION ENTRY</td>
</tr>
<tr>
<td></td>
<td>STARTRTS EQU X'03'</td>
<td></td>
<td>REQUEST-TO-SEND INFORMATION ENTRY</td>
</tr>
<tr>
<td>000003</td>
<td>STAFILG1 DS X</td>
<td></td>
<td>STATUS ENTRY FLAGS</td>
</tr>
<tr>
<td></td>
<td>STACNV EQU X'80'</td>
<td></td>
<td>DATA IS IN CA MODE</td>
</tr>
<tr>
<td>000004</td>
<td>STACNVID DS XL4</td>
<td></td>
<td>CONVID OF CONVERSATION</td>
</tr>
<tr>
<td>000008</td>
<td>STATOTAL DS XL4</td>
<td></td>
<td>TOTAL DATA AVAILABLE (NORM &amp; EXPD)</td>
</tr>
<tr>
<td>00000C</td>
<td>STACURLL DS XL4</td>
<td></td>
<td>CURRENTLY ACTIVE LL FIELD (NORM),</td>
</tr>
<tr>
<td></td>
<td>STACURLR DS XL2</td>
<td></td>
<td>RESERVED (EXPED &amp; RTS_RCVD)</td>
</tr>
<tr>
<td>00000E</td>
<td>STACURLR DS XL2</td>
<td></td>
<td>CURRENT LL REMAINDER (NORM),</td>
</tr>
<tr>
<td></td>
<td>STACURLR DS XL2</td>
<td></td>
<td>RESERVED (EXPED &amp; RTS_RCVD)</td>
</tr>
<tr>
<td>000010</td>
<td>STATENTE DS 0X</td>
<td></td>
<td>END OF STATUS ENTRY</td>
</tr>
</tbody>
</table>

**Feedback code data structure (ISTUSFBC)**

<table>
<thead>
<tr>
<th>LOC</th>
<th>SOURCE STATEMENT</th>
<th>DSECT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>ISTUSFBC DSECT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPLRTNCD RTNCD (FEEDBACK CODE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPLFDB2 FDBK2 (REASON CODE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPLFDB3 FDBK (DATA FLAGS)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPL6RCPR PRIMARY RETURN CODE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPL6RCSC SECONDARY RETURN CODE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* THE FOLLOWING CODES ARE STORED IN EITHER 'RPLRTNCD', OR 'RPLFDB2' OR 'RPLFDB3'. SEE THE INTRODUCTORY COMMENTS FOR EACH GROUP FOR FURTHER INFORMATION. *

* THE FEEDBACK CODE EQUATES ARE AS FOLLOWS: *

USFAOK EQU X'00' NORMAL COMPLETION/CONDITIONAL COMPLETION
USFXORDC EQU X'04' EXTRAORDINARY COMPLETION
USFRESSU EQU X'08' REISSUE THIS REQUEST
USFDAMGE EQU X'0C' DAMAGE - INTEGRITY OF REQUEST/DEVICE
USFENVER EQU X'10' ENVIRONMENT ERROR
USFLOGIC EQU X'14' USER LOGIC ERROR
USFRLGIC EQU X'18' USER LOGIC ERROR - SETONLY IN REG ZERO
USF6CHEK EQU X'20' RPL/RPL6 IN WRONG STATE - SET ONLY IN REG00
USF6WRCK EQU X'24' WRONG CHECK MACRO ISSUED - SET ONLY IN

DSECTs 589
RPLFDB2 CONTAINS A REASON CODE. THIS REASON CODE INDICATES ADDITIONAL INFORMATION ABOUT THE FEEDBACK CODE.

****** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'00' ******

USFAOOK EQU X'00'  OPERATION SUCCESSFULLY COMPLETED
USFRCWNP EQU X'01'  RESET CONDITIONAL WAS NO-OPED
USFRCORDR EQU X'02'  RESET CONDITIONAL SUCCESSFUL - READ-AHEAD DATA PRESENT
USFYTCTN EQU X'03'  YIELDED TO CONTENTION
USFYTYCTL EQU X'04'  YIELDED TO CONTENTION, ERROR LOCK SET
USFATSFI EQU X'05'  AREA TOO SMALL FOR INQUIRE/INTERPRET
USFNOIN EQU X'06'  NO INPUT AVAILABLE
USFIIINA EQU X'07'  INQUIRE INFORMATION NOT AVAILABLE
USFDSTIU EQU X'08'  DESTINATION IN USE
USFNLGFA EQU X'09'  NO LOGON FOUND FOR ACCEPT MATCH
USFANC EQU X'0A'
USF6APPC EQU X'0B'  INDICATES THAT AN ERROR OCCURRED RUNNING APPC, AND REFER TO THE RPL6 PRIMARY AND SECONDARY RETURN CODES
USFINQPS EQU X'0D'  MORE SESSIONS PENDING RECOVERY ON

****** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'04' ******

USFRVIRC EQU X'00'  RVI RECEIVED, ERROR LOCK SET
USFATNRC EQU X'01'  ATTENTION RECEIVED, ERROR LOCK SET
USFSBCSM EQU X'02'  BSC STATUS MSG RECEIVED
USFEXRQ EQU X'03'  EXCEPTION REQUEST RECEIVED
USFEXRS EQU X'04'  EXCEPTION RESPONSE RECEIVED
USFNQN EQU X'05'  RESOURCE KNOWN AS NQN ONLY

****** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'08' ******

USFSTALF EQU X'00'  TEMPORARY OUT OF STORAGE SITUATION EXISTS RPL ECB/EXIT NOT POSTED/INVOKED
USFUOEDU EQU X'00'  I/O ERROR, DEVICE STILL USABLE ER LK SET
USFDVUNS EQU X'01'  I/O ERROR, DEVICE NOT USABLE ER LCK SET
USFUNTRK EQU X'02'  REQUEST RESET BY TEST REQUEST MESSAGE
USFBTHEX EQU X'03'  BUFFER THRESHOLD EXCEEDED
USFBTEOR EQU X'04'  BUF THRESHOLD EXCEEDED, ONLY READS ALLOW
USFNCPAO EQU X'05'  NCP ABENDED, RESTART O.K.
USFLIORP EQU X'06'  LAST I/O REQUEST PURGED
USFRECIP EQU X'07'  LAST I/O REQUEST PURGED
USFREJET EQU X'08'  REQUEST TERMINAL RESTARTED AFTER FAILURE
USFDQPCO EQU X'09'  QUEUE OPNDST CANCELLED BY CLSDST
USFDRESR EQU X'0A'  REQUEST RESET BY THE USER
USFCHLCC EQU X'0B'  CLSDST OR TERMSESS ISSUED OR UNBIND SENT

****** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'0C' ******

USFIOEDU EQU X'00'  I/O ERROR, DEVICE STILL USABLE ER LK SET
USFDVUNS EQU X'01'  I/O ERROR, DEVICE NOT USABLE ER LCK SET
USFUNTRK EQU X'02'  REQUEST RESET BY TEST REQUEST MESSAGE
USFBTHEX EQU X'03'  BUFFER THRESHOLD EXCEEDED
USFBTEOR EQU X'04'  BUF THRESHOLD EXCEEDED, ONLY READS ALLOW
USFNCPAO EQU X'05'  NCP ABENDED, RESTART O.K.
USFLIORP EQU X'06'  LAST I/O REQUEST PURGED
USFRECIP EQU X'07'  LAST I/O REQUEST PURGED
USFREJET EQU X'08'  REQUEST TERMINAL RESTARTED AFTER FAILURE
USFDQPCO EQU X'09'  QUEUE OPNDST CANCELLED BY CLSDST
USFDRESR EQU X'0A'  REQUEST RESET BY THE USER
USFCHLCC EQU X'0B'  CLSDST OR TERMSESS ISSUED OR UNBIND SENT

****** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'10' ******

USFTANAV EQU X'00'  TERMINAL OR APPLICATION NOT AVAILABLE
USFSBFAL EQU X'01'  SESSION BIND FAILED
USFTAPUA EQU X'02'  TARGET APPLICATION UNACCEPTABLE
USFVTMLV EQU X'03'  VTAM IS HALTING
USFILRS EQU X'04'  INCOMPATIBLE DEFINITION
USFPCF EQU X'05'  PERMANENT FAILURE IN PATH
USFANS EQU X'06'  AUTO NETWORK SHUTDOWN
USFVFOC EQU X'07'  VARY DEACTIVATE IMMEDIATE OCCURRED
USFDISCO EQU   X'08'          DISCONNECT OCCURRED
USFUTSCR EQU   X'09'          UNCONDITIONAL TERMINATE SELF CMD RECEIVED
USFSYERR EQU   X'0A'          APPARENT VTAM ERROR
USFDIDOL EQU   X'0B'          DISCONNECT ON DIAL-OUT LINE
USFDIDIL EQU   X'0C'          DISCONNECT ON DIAL-IN LINE
*    NOTE - X'0D' AND X'0E' - RPL ECB/EXIT NOT POSTED/INVOKED         *
USFVTMNA EQU   X'0D'          VTAM INACTIVE FOR THAT APPLICATION
USFABNDO EQU   X'0E'          ABEND CONDITION HAS OCCURRED
*                                                                     *
USFVTBFO EQU   X'0F'          VTAM BUFFER OVERFLOW
USFTERM EQU   X'10'          CONDITIONAL TERM SELF
USFOSDTF EQU   X'11'          SDT FAILURE ON OPNDST
USFMFF   EQU   X'12'          MACRO FUNCTION FAILED,SENSE INCLUDED
USF6APRJ EQU   X'13'          ATTEMPT TO START 6.2 SESSION: REQUEST
*                              REJECTED                           *
USF6APST EQU   X'14'          ATTEMPT TO START 6.2 SESSION: PENDING
*                              SESSION TERMINATED                *
USF6APIS EQU   X'15'          MUST ISSUE APPCCMD
USFNONSW EQU   X'16'          SWITCHED OPERATION ATTEMPTED ON
*                              NONSWITCHED DEVICE                *
USFVTMNA EQU   X'17'          VTAM INACTIVE FOR THAT APPLICATION
USFABNDO EQU   X'18'          ABEND CONDITION HAS OCCURRED
*                                                                     *
USFVTBFO EQU   X'19'          VTAM BUFFER OVERFLOW
USFCRPLN EQU   X'1A'          CHECKED RPL IS NOT ACTIVE
*                        ONLY OCCURS FOLLOWING A CHECK MACRO REQUEST  *
USFIDA   EQU   X'1B'          INVALID DATA AREA OR LENGTH
USFILDOA EQU   X'1C'          INVALID LDO ADDRESS
USFJTOJ  EQU   X'1D'          JUMP LDO TO JUMP
USFMT100 EQU   X'1E'          MORE THAN 100 LDOS
USFRMD32 EQU   X'1F'          READ MODIFIED TO NON-3270 DEVICE
*    *****    REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'14' *****    *
*    *    USFNONVR EQU   X'00'          RPL CONTAINS A NON-VTAM REQUEST CODE
*                              RPL ECB/EXIT NOT POSTED/INVOKED         *
USFNOTAS EQU   X'01'     NOT ASSIGNED
*    USFXEEXTA2 EQU   X'02'          RPL INDICATES EXIT, EXIT ADDR IS ZERO
*                              RPL ECB/EXIT NOT POSTED/INVOKED         *
USFXEXTZ EQU   X'03'          RPL IND EXTERNAL ECB, ECB ADDR IS ZERO
*                              RPL ECB/EXIT NOT POSTED/INVOKED         *
USFCRPNL EQU   X'04'          CHECKED RPL IS NOT ACTIVE
*                              ONLY OCCURS FOLLOWING A CHECK MACRO REQUEST  *
USFCBERR EQU   X'10'          RPL POINTS TO INVALID ACB
USFRNORT EQU   X'11'          NO RTYPE SPECIFIED
USFCLSIP EQU   X'12'          CLSDST IN PROGRESS
USFCIDNG EQU   X'13'          CID IS INVALID
USFILLDP EQU   X'14'          LDO COMMAND FIELD IS INVALID
USFMANCR EQU   X'15'          READ NOT CHAINED
USFST00D EQU   X'16'          SOLICIT SPECIFIC TO OUTPUT ONLY DEVICE
USFRTOOD EQU   X'17'          READ TO OUTPUT ONLY DEVICE
USFWTOI EQU   X'18'          WRITE TO INPUT ONLY DEVICE
USFENWS EQU   X'19'          ERASE TO INVALID DEVICE
USFCELE3 EQU   X'1A'          WRITE EAU TO NON-3270
USFCWTOO EQU   X'1B'          WRITE CONV TO OUTPUT ONLY DEVICE
USFCTN32 EQU   X'1C'          COPY TO NON-3270 DEVICE
USFRCINV EQU   X'1D'          INVALID DATA AREA OR LENGTH
USFILDOA EQU   X'1E'          INVALID LDO ADDRESS
USFJA0B EQU   X'1F'          JUMP LDO TO JUMP
USFRMD32 EQU   X'20'          MORE THAN 100 LDOS
USFIREST EQU   X'21'          READ MODIFIED TO NON-3270 DEVICE
USFSRNOE EQU   X'22'          RESTART LDO IS NOT ALONE
USFMCINV EQU   X'23'          INVALID MACRO REQUEST TYPE
USFSIDE EQU   X'24'          ASID MISMATCH
USFSWBLK EQU   X'25'          WRITE ERASE BLOCK
USF6C5DC EQU   X'26'          SOLICIT LDO WITH DATA CHAINING
USFIREST EQU   X'27'          RESET OPTION CODE INVALID
USFWB32 EQU   X'28'          WRITE BLOCK TO 3270 DEVICE
USFRMD32 EQU   X'29'          READ MODIFIED TO NON-3270 DEVICE
USFCTN32 EQU   X'2A'          COPY TO NON-3270 DEVICE
USFWCNNR EQU   X'2B'          WRITE CONV ISSUED WHEN DATA EXPECTED
USFRNFT3 EQU   X'2C'          OUTPUT NOT PRECEDED BY INPUT
USF6C5CNV EQU   X'2D'          RESTART LDO IS NOT ALONE
USFINVME EQU   X'2E'          INVALID READ MODE
USF6C5GNT EQU   X'2F'          EXCESSIVE LEADING GRAPHICS, ERROR LK SET
USFPCPN EQU   X'30'          COPY COUNT ERROR
USFIDEA EQU   X'31'          INVALID DATA AREA OR LENGTH, ERROR LK SET
USFUSELE EQU   X'32'          REQUEST INVALID FOR DEVICE, ERROR LK SET
USFCRNF EQU   X'33'          CONV. REPLY NOT POSSIBLE, ERROR LOCK SET
USFNOR EQU   X'34'           NO READ WHERE REQUIRED, ERROR LOCK SET
USFCY2E EQU   X'35'          COPY WRONG CLUSTER, ERROR LOCK SET
USFRELNP EQU   X'36'          REQUEST LOCK NOT ALLOWED, ERROR LOCK SET
** PENDING ACTIVE STATE

USFPARML EQU X'82'  PARAMETER LENGTH INVALID
USFSFERR EQU X'83'  SUBFIELD NOT SUPPORTED, INVALID
*  COMBINATION OF SUBFIELDS, OR
*  SUBFIELD FORMAT ERROR
USFASDAZ EQU X'84'  ZERO NIBASDPA FIELD
USFSMBRS EQU X'85'  SESSION IS IN RECOVERY STATE AND MUST BE
*  RESTORED
USFSESSA EQU X'86'  SESSIONS OR AFFINITIES EXIST
USFNAME EQU X'87'  RESOURCE NAME AND GENERIC NAME EQUAL
USFNSOPT EQU X'88'  NO SPT EXISTS
USFNSECM EQU X'89'  NO SECURITY AUTHORIZATION FOR GENERIC
*  RESOURCE
USFDIFNM EQU X'8A'  ALREADY REGISTERED WITH A DIFFERENT
*  GENERIC NAME
USFNOMAP EQU X'8B'  NOT REGISTERED AS A GENERIC RESOURCE
*  NETWORK ID
USFCPNAM EQU X'8C'  MAPPING ALREADY EXISTS ON A DIFFERENT
*  SYSPLEX NODE
USFCNFAC EQU X'8D'  CONFLICTING APPC CAPABILITY
USVFVTAMO EQU X'8E'  SPTE IS OWNED BY VTAM
USFUSVAR EQU X'8F'  GENERIC NAME CONFLICTS WITH AN
*  EXISTING USERVAR
USFGENUJA EQU X'90'  TSO GENERIC NAME CONFLICT
USFGGMNA EQU X'91'  SETLOGON GNAMESUB FAILURE
USFSTKNV EQU X'92'  STOKEN NOT VALID

***** NO REASON CODE EQUATES EXIST FOR RPLRTNCD EQUALS X'18' *****

***** EQUATES FOR RPLFDB3 ON RETURN FROM INQUIRE IF RPLRTNCD IS X'00' *****

USFIACT EQU X'00'  APPLICATION IS ACTIVE
USFIINA EQU X'04'  APPLICATION IS INACTIVE
*  SEE USFANC (X'0A') UNDER RPLFDB2
*  WHEN RTNCD = X'00'
USFINA EQU X'08'  APPLICATION WILL NOT ACCEPT LOGONS
USFITNA EQU X'0C'  APPLICATION IS TEMPORARILY NOT
*  ACCEPTING LOGONS
USFIQUIE EQU X'10'  APPLICATION IS QUIESCING
USFILACT EQU X'80'  RESOURCE IS ACTIVE
USFILINA EQU X'84'  RESOURCE IS NOT ACTIVE

***********************************************************************
*** THE FOLLOWING ARE ALL THE RPL6RCPR (PRIMARY RETURN
*** CODE) VALUES FOR APPC/VTAM.
***
***
USF6OK EQU X'0000'  OK
USF6ALLC EQU X'0004'  ALLOCATION ERROR
USF6CNSA EQU X'0008'  CNOS ALLOCATION ERROR
USF6CNSN EQU X'000C'  CNOS RESOURCE FAILURE, NO RETRY
USF6CRRJ EQU X'0010'  COMMAND RACE REJECT
USF6DABP EQU X'0014'  DEALLOCATE ABEND PROGRAM
USF6DABS EQU X'0018'  DEALLOCATE ABEND SERVICE
USF6DABT EQU X'001C'  DEALLOCATE ABEND TIMER
USF6CNSR EQU X'0020'  CNOS FAILURE, RETRY
USF6LRBE EQU X'0024'  LOGICAL RECORD BOUNDARY ERROR
USF6LCL EQU X'0028'  LU MODE SESSION LIMIT CLOSED
USF6PARM EQU X'002C'  PARAMETER ERROR
USF6PENT EQU X'0030'  PROGRAM ERROR NO TRUNCATION
USF6PEPU EQU X'0034'  PROGRAM ERROR PURGING
USF6PETR EQU X'0038'  PROGRAM ERROR TRUNCATING
USF6SENT EQU X'003C'  SERVICE ERROR NO TRUNCATION
USF6SEPUP EQU X'0040'  SERVICE ERROR PURGING
USF6SETR EQU X'0044'  SERVICE ERROR TRUNCATING
USF6SRFR EQU X'0048'  RESOURCE FAILURE, NO RETRY
USF6SRFR EQU X'004C'  RESOURCE FAILURE, RETRY
USF6SRER EQU X'0050'  STATE ERROR
USF6URMD EQU X'0054'  UNRECOGNIZED MODE NAME
USF6UNSC EQU X'0058'  UNSUCCESSFUL, SESSION NOT AVAILABLE
*  USF6UECR EQU X'005C'  USER ERROR CODE RECEIVED
USF6NOMF EQU X'0060'  NO FMHS AVAILABLE
USF6ACFL EQU X'0064'  ACTIVATION FAILURE
USF6SLEX EQU X'0068'  LU MODE SESSION LIMIT EXCEEDED
USF6SACT EQU X'006C'  SESSION NOT PENDING
USF6STOR EQU X'0070'  TEMPORARY STORAGE SHORTAGE OR RESOURCE
SHORTAGE
USF6HALT EQU X'0074'        HALT ISSUED
USF6VIYA EQU X'0078'        VTAM INACTIVE FOR YOUR ACB
USF6RQAB EQU X'007C'        REQUEST ABORTED
USF6DLNR EQU X'0080'        DEALLOCATE NORMAL
USF6STSH EQU X'0084'        STORAGE SHORTAGE
USF6CREJ EQU X'0088'        CANCELLED BY REJECT OR DEALLOCATE ABND*
* USF6PROE EQU X'008C'        PARTNER COMMITTED PROTOCOL VIOLATION
* USF6NOTA EQU X'0090'        APPLICATION NOT APPC CAPABLE
* USF6SDRJ EQU X'0094'        SEND DATA REJECTED INVALID STATE
* USF6STGS EQU X'0098'        STORAGE SHORTAGE WHILE SENDING DATA
* USF6RNAL EQU X'00A0'        REQUEST NOT ALLOWED
* USF6SPMD EQU X'00A4'        MODE MUST BE RESTORED BEFORE USING
* USF6INER EQU X'00A8'        INTERNAL VTAM ERROR
* USF6RSUN EQU X'00A9'        RESTORE UNNECESSARY - NO SESSIONS TO RESTORE
* USF6RSIN EQU X'00AB'        RESTORE INCOMPLETE - INPUT WORK AREA TOO SMALL
* USF6NINA EQU X'00AC'        NO IMMEDIATELY AVAILABLE INFORMATION FOR REQUEST
* USF6RTEC EQU X'00AD'        REQUEST TERMINATED BY END OF CONVERSATION
* USF6ANMS EQU X'00AE'        SESSIONS WILL USE APPL NETWORK NAME, GENERIC NAME WAS REQUESTED
* USF6GNMS EQU X'00AF'        SESSIONS WILL USE GENERIC NAME, APPL NETWORK NAME WAS REQUESTED
* USF6NAM1 EQU X'00B0'        AS SPECIFIED, PARTNER LU KNOWN BY DIFFERENT NAME
* USF6NAM2 EQU X'00B1'        AS NEGOTIATED, PARTNER LU KNOWN BY DIFFERENT NAME
* USF6ALNR EQU X'00B2'        ALLOCATION FAILURE, NO RETRY
* USF6ALR  EQU X'00B3'        ALLOCATION FAILURE, RETRY
* USF6ALCM EQU X'00B4'        CONVERSATION TYPE MISMATCH
* USF6ALPI EQU X'00B5'        PIP NOT ALLOWED
* USF6ALPP EQU X'00B6'        PIP NOT SPECIFIED CORRECTLY
* USF6ALSC EQU X'00B7'        SECURITY NOT VALID
* USF6ALSX EQU X'00B8'        SYNC LEVEL NOT SUPPORTED BY LU
* USF6ALSL EQU X'00B9'        SYNC LEVEL NOT SUPPORTED BY PROGRAM
* USF6ALT P EQU X'00BA'        TPN NOT RECOGNIZED
* USF6ALT N EQU X'00BB'        TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY
* USF6ALTR EQU X'00BC'        TRANSACTION PROGRAM NOT AVAILABLE, RETRY
* USF6ALRN EQU X'00BD'        CANNOT RECONNECT TRANSACTION PROGRAM, NO RETRY
* USF6ALRR EQU X'00BE'        CANNOT RECONNECT TRANSACTION PROGRAM, RETRY
* USF6ALS N EQU X'00BF'        RECONNECT NOT SUPPORTED BY PROGRAM
* USF6SPMA EQU X'00C0'        MODE MUST BE RESTORED BEFORE USING
* USF6DAQ EQU X'00C1'        DEALLOCATION REQUESTED
* USF6ALSF EQU X'00C2'        REQUESTED SYNC LEVEL NOT ALLOWED
* USF6LNSF EQU X'00C3'        LU PAIR NOT SUPPORTING FULL-DUPLEX CONVERSATIONS
*
*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'0000' (USF6OK).
*** USF6OKSC EQU X'0000'        OK
USF6ASSP EQU X'0001'        AS SPECIFIED
USF6ASNG EQU X'0002'        AS NEGOTIATED
USF6RCVR EQU X'0003'        RECEIVE SPECIFIC REJECTED
USF6SNGL EQU X'0004'        PARTNER LU SUPPORTS SINGLE SESSION
* USF6INER EQU X'0005'        INTERNAL VTAM ERROR
* USF6RSUN EQU X'0006'        RESTORE UNNECESSARY - NO SESSIONS TO RESTORE
* USF6RSIN EQU X'0007'        RESTORE INCOMPLETE - INPUT WORK AREA TOO SMALL
* USF6NINA EQU X'0008'        NO IMMEDIATELY AVAILABLE INFORMATION FOR REQUEST
* USF6RTEC EQU X'0009'        REQUEST TERMINATED BY END OF CONVERSATION
* USF6ANMS EQU X'000A'        SESSIONS WILL USE APPL NETWORK NAME, GENERIC NAME WAS REQUESTED
* USF6GNMS EQU X'000B'        SESSIONS WILL USE GENERIC NAME, APPL NETWORK NAME WAS REQUESTED
* USF6NAM1 EQU X'000C'        AS SPECIFIED, PARTNER LU KNOWN BY DIFFERENT NAME
* USF6NAM2 EQU X'000D'        AS NEGOTIATED, PARTNER LU KNOWN BY DIFFERENT NAME
* USF6ALNR EQU X'000E'        ALLOCATION FAILURE, NO RETRY
* USF6ALR  EQU X'000F'        ALLOCATION FAILURE, RETRY
* USF6ALCM EQU X'0010'        CONVERSATION TYPE MISMATCH
* USF6ALPI EQU X'0011'        PIP NOT ALLOWED
* USF6ALPP EQU X'0012'        PIP NOT SPECIFIED CORRECTLY
* USF6ALSC EQU X'0013'        SECURITY NOT VALID
* USF6ALSX EQU X'0014'        SYNC LEVEL NOT SUPPORTED BY LU
* USF6ALSL EQU X'0015'        SYNC LEVEL NOT SUPPORTED BY PROGRAM
* USF6ALT P EQU X'0016'        TPN NOT RECOGNIZED
* USF6ALT N EQU X'0017'        TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY
* USF6ALTR EQU X'0018'        TRANSACTION PROGRAM NOT AVAILABLE, RETRY
* USF6ALRN EQU X'0019'        CANNOT RECONNECT TRANSACTION PROGRAM, NO RETRY
* USF6ALRR EQU X'001A'        CANNOT RECONNECT TRANSACTION PROGRAM, RETRY
* USF6ALS N EQU X'001B'        RECONNECT NOT SUPPORTED BY PROGRAM
* USF6SPMA EQU X'001C'        MODE MUST BE RESTORED BEFORE USING
* USF6DAQ EQU X'001D'        DEALLOCATION REQUESTED
* USF6ALSF EQU X'001E'        REQUESTED SYNC LEVEL NOT ALLOWED
* USF6LNSF EQU X'001F'        LU PAIR NOT SUPPORTING FULL-DUPLEX CONVERSATIONS
*
THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'0008' (USF6CNSA).

- USF6CANR EQU X'0000' ALLOCATION FAILURE, NO RETRY
- USF6CAR EQU X'0001' ALLOCATION FAILURE, RETRY
- USF6CATR EQU X'0002' TRANSACTION PROGRAM NOT AVAILABLE, RETRY
- USF6CATN EQU X'0003' TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY
- USF6CACM EQU X'0004' CONVERSATION TYPE MISMATCH
- USF6CASC EQU X'0005' SECURITY NOT VALID
- USF6SPMC EQU X'0006' MODE MUST BE RESTORED BEFORE USING
- USF6NQNM EQU X'0007' NETWORK QUALIFIED NAME MISMATCH

THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'10' (USF6CRRJ).

- USF6CRPR EQU X'0000' PARTNER GRANTED RETRY
- USF6CRLR EQU X'0001' CONTROL OPERATOR OF LOCAL LU RETRIED
- USF6PCIP EQU X'0002' PARTNER CNOS IN PROGRESS
- USF6LPSS EQU X'0003' LU IS IN PENDING SINGLE STATE
- USF6PLSS EQU X'0004' PARTNER LU STARTING SESSION

THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'002C' (USF6PARM).

- USF6IVLU EQU X'0000' INVALID LU NAME OR NETID
- USF6IVMD EQU X'0001' INVALID MODE
- USF6IVCI EQU X'0002' INVALID CONVERSATION ID
- USF6IVLL EQU X'0003' INVALID LL
- USF6IVSV EQU X'0004' INVALID VALUES FOR SNASVCMG MODE
- USF6SNAR EQU X'0006' SNASVCMG MODE CANNOT CURRENTLY BE RESET
- USF6MMEX EQU X'0007' MINWINL PLUS MINWINR EXCEEDS SESSLIM
- USF6LNIN EQU X'0008' SUPPLIED LENGTH INSUFFICIENT
- USF6NSL EQU X'0009' INCOMPLETE SESSION LIMITS STRUCTURE
- USF6INFM EQU X'000A' INCOMPLETE FMHS SUPPLIED
- USF6INGD EQU X'000B' INCOMPLETE GDS VARIABLE SUPPLIED
- USF60EXT EQU X'000C' ZERO EXIT FIELD
- USF60ECB EQU X'000D' ZERO ECB FIELD
- USF60IAS EQU X'000E' REQUEST INVALID FOR ADDRESS SPACE
- USF6CBIN EQU X'000F' CONTROL BLOCK INVALID
- USF6INDL EQU X'0010' INVALID DATA ADDRESS OR LENGTH
- USF6PRVO EQU X'0011' PREVIOUS MACRO INSTRUCTION OUTSTANDING
- USF6BLIV EQU X'0012' BUFFER LIST LENGTH INVALID
- USF6NOMD EQU X'0013' NO CORRESPONDING MODE IN LM TABLE
- USF6IVBP EQU X'0014' INVALID BIND PARAMETERS
- USF6IVTP EQU X'0015' INVALID TPN
- USF6NOLU EQU X'0016' NO CORRESPONDING LU IN LM TABLE
- USF6IMDF EQU X'0017' INVALID MODE SPECIFIED
- USF6ILSP EQU X'0018' INVALID LIMIT SPECIFIED
- USF6SMAI EQU X'0019' SNASVCMG MODE ALREADY INITIALIZED
- USF6ALLS EQU X'001A' ALL MODES SPECIFIED ON SINGLE SESSION LU
- USF6SMSS EQU X'001B' SNASVCMG MODE FOR SINGLE SESSION LU
- USF6SSMI EQU X'001C' SINGLE SESSION, MODE ALREADY INITIALIZED
- USF6IDID EQU X'001E' CID INVALID
- USF6APNA EQU X'001F' APPCCMD ISSUED FOR NON-APPC
- USF6PRRO EQU X'0020' PREVIOUS REJECT REQUEST OUTSTANDING
- USF6DARJ EQU X'0021' DEALLOCATE ABND* REJECTED, RETRY
- USF6DIVC EQU X'0022' INVALID CONTROL OR QUALIFY VALUE
- USF6INSI EQU X'0023' INVALID SESSION INSTANCE IDENTIFIER
- USF6PSHI EQU X'0024' PS HEADER NOT SUPPLIED
- USF6PSLI EQU X'0025' PS HEADER LENGTH INSUFFICIENT
- USF6NMSC EQU X'0026' SESSION INSTANCE IDENTIFIER AND
- USF6IDET EQU X'0027' INVALID DEACTIVATION TYPE CODE
- USF6NCRY EQU X'0028' CRYPTOGRAPHY NOT ALLOWED ON MODE
USF6INLI EQU X'0029'    INVALID LIST VALUE SPECIFIED ON APPCCMD FOR RESTORE
USF6INCG EQU X'002A'    INVALID CGID VALUE ON ALLOCATE
USF6NONI EQU X'002B'    NETWORK QUALIFIED NAME REQUIRED
USF6NIEL EQU X'002C'    INVALID EXPEDITED DATA LENGTH SPECIFIED
USF6INSC EQU X'002D'    INVALID SENSE CODE SPECIFIED
USF6VANV EQU X'002E'    VECTOR AREA NOT VALID
USF6VALI EQU X'002F'    VECTOR AREA LENGTH INSUFFICIENT
USF6STNV EQU X'0030'    STORAGE TYPE NOT VALID
USF6VALS EQU X'0031'    SENDRCV SPECIFIED WITHOUT OPTCD=BUFFLST|XBUFLST
USF6UNKV EQU X'0032'    UNEXPECTED VECTOR PROVIDED ON APPCCMD
USF6VPV EQU X'0033'    A REQUIRED VECTOR WAS NOT PROVIDED OR SPECIFIED INCORRECTLY
USF6LNSP EQU X'0034'    PASSWORD SUBSTITUTION VALUE SET IN ERROR

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'005C' (USF6UECR).

USF6FNGR EQU X'0000'    FOLLOWING NEGATIVE RESPONSE
USF6WNGR EQU X'0001'    WITHOUT NEGATIVE RESPONSE

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'009C' (USF6RSTF).

USF6LSLR EQU X'0001'    RESTORE ISSUED BEFORE SETLOGON START

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'00A0' (USF6RNAL).

USF6LNSE EQU X'0001'    LU PAIR NOT SUPPORTING EXPEDITED DATA REQUESTS
USF6RQBL EQU X'0002'    REQUEST BLOCKED DUE TO PENDING CONVERSATION TERMINATION
USF6RNX EQU X'0003'    EXECUTION OF REQUEST TERMINATED CONTROL/QUALIFY VALUE INVALID ON FULL-DUPLEX CONVERSATION
USF6RNF EQU X'0004'    EXPEDITED DATA RESPONSE OUTSTANDING FUNCTION RESERVED
USF6ENEL EQU X'0005'    NAMED RESOURCE NOT ELIGIBLE FOR REQUESTED ALTERATION

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'00A8' (USF6ENVE).

USF6OSLV EQU X'0000'    OPERATING SYSTEM LEVEL DOES NOT SUPPORT REQUESTED FUNCTION
USF6XMS EQU X'0001'    SUSPEND FAILURE
USF6XMR EQU X'0002'    RESUME FAILURE

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'00B0' (USF6ERIN).

USF6EIAP EQU X'0001'    DEALLOCATE ABEND PROGRAM
USF6EIAS EQU X'0002'    DEALLOCATE ABEND SERVICE
USF6EIAT EQU X'0003'    DEALLOCATE ABEND TIMER
USF6EIAD EQU X'0004'    ALLOCATION ERROR
USF6EIUN EQU X'0005'    UNKNOWN TERMINATION TYPE RECEIVED
USF6EIRR EQU X'0006'    RESOURCE FAILURE, RETRY
USF6EIIN EQU X'0007'    RESOURCE FAILURE, NO RETRY

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'00B8' (USF6NER).
APPCCMD vector lists (ISTAPCVL)

LOC SOURCE STATEMENT
*** MAPPING FOR VECTORLIST HEADER (LENGTH FIELD) **
000000 ISTAPCVA DSECT VECTOR LIST
*
000001 APCVALEN DS HL2 LENGTH OF VECTOR LIST
   (INCLUDES LENGTH FIELD & VECTORS)
000002 APCVADTA DS 0X VECTORS
*
***********************************************************************
*** GENERALIZED MAPPING FOR EXAMINING OR BUILDING COMMON FIELDS IN **
*** ALL APPCCMD VECTORS IN THE VECTOR LISTS POINTED TO BY RPL6VAIA **
*** AND RPL6VAIA **
***********************************************************************
000000 ISTAPCVT DSECT VECTOR TEMPLATE
000000 APCVTLEN DS HL2 VECTOR LENGTH
000002 APCVTKEY DS X VECTOR KEY
000003 APCVTDTA DS 0X VECTOR DATA
*
*****************************************************************************
*** VECTORS PASSED FROM VTAM TO APPLICATION AT APPCCMD COMPLETION **
*** Note: Highorder bit in vector key is off for all vectors sent **
*** from VTAM to application. **
*** **
*****************************************************************************
*** ISTAPC10 - maps the VTAM-to-APPL Required INFORMATION vector. **
*** - Returned on all APPCCMD macros if a vector area is **
*** provided. **
*** - Indicates whether VTAM was able to return vector **
*** information successfully and length needed. **
*** - NOTE: Application-provided vector area must be large **
*** enough to accept at least this vector. **
*****************************************************************************
000000 ISTAPC10 DSECT INFORMATION VECTOR
*
000000 APC10LEN DS HL2 VECTOR LENGTH
000002 APC10KEY DS X VECTOR KEY
000003 APC10KYC EQU X'10' VECTOR KEY X'10'
000003 APC10DTA DS 0X VECTOR DATA FIELDS
000003 APC10FLG DS X FLAGS
000003 APC10IVAL EQU X'80' INSUFFICIENT VECTOR AREA LENGTH
000004 DS X RESERVED
000005 DS X RESERVED
000006 APC10VLN DS HL2 VECTOR AREA LENGTH NEEDED
*
*****************************************************************************
*** ISTAPC12 - Maps the Partner’s DCE Capability vector. **
*** - Returned on these APPCCMD completions if DCE is **
*** active: **
*** APPCCMD CONTROL=PREALLOC **
*** APPCCMD CONTROL=RDVFMS **
*** APPCCMD CONTROL=OPRCNTL QUALIFY=CNOS **
*** APPCCMD CONTROL=OPRCNTL QUALIFY=DISPLAY **
*** - Also returned on ATTN(CNOS) if DCE is active. **
*** - Contains the Security Mechanisms Data subfield **
*** exchanged during BIND processing if DCE is active.  **

000000 ISTAPC12 DSECT PARTNER'S DCE CAPABILITY VECTOR
   * MAPPING
000000 APC12LEN DS HL2 LENGTH OF VECTOR (INCLUDING
   * LENGTH OF THIS FIELD)
000002 APC12KEY DS X VECTOR KEY
APC12KEY EQU X'12' VECTOR KEY X'12'
000003 APC12OTA DS 0X SECURITY MECHANISMS DATA
APC12OTA EQU X'01' DCE AUTHENTICATION
APC12OTA EQU X'02' KRYPTOKNIGHT
APC12OTA EQU X'03' KERBEROS V5
APC12OTA EQU X'04' DCE PERFORMANCE MECHANISM
*
*
***********************************************************************
*** ISTAPC13 - maps the LOCAL NONCE vector.                         **
*** - Returned for these APPCCMD completions if                      **
***    password substitution is supported on session:                **
***    APPCCMD CONTROL=PREALLOC                                     **
*** - Contains random data used for password                       **
***    substitution.                                                 **
***********************************************************************
000000 ISTAPC13 DSECT MAPPING FOR LOCAL NONCE VECTOR
000000 APC13LEN DS HL2 LENGTH OF VECTOR
000002 APC13KEY DS X VECTOR KEY
APC13KEY EQU X'13' KEY IS X'13'
000003 APC13OTA DS 0X NONCE DATA
000003 DS XL1 RESERVED
000004 APC13NOF DS CL8 NONCE FIELD
*
*
***********************************************************************
*** ISTAPC14 - maps the PARTNER'S NONCE vector.                      **
*** - Returned for these APPCCMD completions if                      **
***    password substitution is supported on session:                **
***    APPCCMD CONTROL=PREALLOC                                     **
*** - Contains random data used for password                       **
***    substitution.                                                 **
***********************************************************************
000000 ISTAPC14 DSECT MAPPING FOR PARTNER NONCE
000000 APC14LEN DS HL2 LENGTH OF VECTOR
000002 APC14KEY DS X VECTOR KEY
APC14KEY EQU X'14' KEY IS X'14'
000003 APC14OTA DS 0X NONCE DATA
000003 DS XL1 RESERVED
000004 APC14NOF DS CL8 NONCE FIELD
*
*
***********************************************************************
*** ISTAPC15 - maps the SEND FMH_5 SEQUENCE NUMBER vector.           **
*** - Returned for these APPCCMD completions if                      **
***    password substitution is supported on session:                **
***    APPCCMD CONTROL=PREALLOC                                     **
*** - Contains the number of FMH_5s which have flowed on             **
***    this session from the partner LU.                            **
***********************************************************************
000000 ISTAPC15 DSECT MAPPING FOR SEND FMH_5
000000 APC15LEN DS HL2 LENGTH OF VECTOR
000002 APC15KEY DS X VECTOR KEY
APC15KEY EQU X'15' KEY IS X'15'
000003 DS XL1 RESERVED
000004 APC15SNF DS 0X SEQUENCE NUMBER FIELD
000004 APC15SNH DS XL4 SEQUENCE NUMBER FIELD -                  **
   * HIGH-ORDER BITS
000008 APC15SLN DS XL4 SEQUENCE NUMBER FIELD -                  **
   * LOW-ORDER BITS
*
*
***********************************************************************
*** ISTAPC16 - maps the RECEIVE FMH_5 SEQUENCE NUMBER vector.        **
*** - Returned for these APPCCMD completions if                      **
***    password substitution is supported on session:                **
*** - Contains the number of FMH_5s which have flowed on             **
***    this session from the partner LU.                            **
***********************************************************************
000000 ISTAPC16 DSECT MAPPING FOR RECEIVE FMH_5
   * SEQUENCE NUMBER VECTOR
******************************************************************************
*** ISTAPC17 - maps the PCID vector.                                   **
***          - Returned for these APPCCMD completions:                 **
***               APPCCMD CONTROL=ALLOC                              **
***               APPCCMD CONTROL=PREALLOC                           **
***               APPCCMD CONTROL=RCVFMS5                            **
***          - Contains the PCID for the session being used by the **
***            conversation.                                         **
******************************************************************************

000000  ISTAPC17 DSECT                      MAPPING FOR PCID VECTOR
000000  APC17LEN DS    HL2                  LENGTH OF VECTOR
000002  APC17KEY DS    X                    VECTOR KEY
APC17KYC EQU   X'17'                KEY IS X'17'
000003  APC17DTA DS    0X                   VECTOR DATA FIELDS
000003  APC17PCF DS    CL8                  SESSION PCID FIELD
*
******************************************************************************
*** ISTAPC18 - maps the NAME CHANGE vector.                           **
***          - Returned for these APPCCMD completions and exits:      **
***               APPCCMD CONTROL=ALLOC                              **
***               APPCCMD CONTROL=OPRCNTL,QUALIFY=CNOS               **
***               APPCCMD CONTROL=PREALLOC                           **
***               ATTN(CNOS) exit                                    **
***            ...when a RCVD_NAME LU entry has been changed to     **
***            a VARIANT_NAME LU entry in the LU-Mode Table.         **
******************************************************************************

000000  ISTAPC18 DSECT                      MAPPING FOR NAME CHANGE VECTOR
000000  APC18LEN DS    HL2                  LENGTH OF VECTOR
000002  APC18KEY DS    X                    VECTOR KEY
APC18KYC EQU   X'18'                KEY IS X'18'
000003  APC18DTA DS    0X                   VECTOR DATA FIELDS
000003  APC18NID DS    CL8                  NETWORK IDENTIFIER OF THE LU
000003  APC18RCV DS    CL8                  LUNAME IN RCVD_NAME LU ENTRY
000003  APC18SUP DS    CL8                  LUNAME IN SUPPLIED_NAME ENTRY
*
******************************************************************************
*** ISTAPC19 - maps the Session Information vector.                    **
***          - Returned for these APPCCMD completions:                 **
***               APPCCMD CONTROL=ALLOC                              **
***               APPCCMD CONTROL=PREALLOC                           **
***               APPCCMD CONTROL=RCVFMS5                            **
***            ...to provide session characteristics information    **
***            for the conversation.                                **
******************************************************************************

000000  ISTAPC19 DSECT                      MAPPING FOR SESSION INFORMATION
000000  APC19LEN DS    HL2                  LENGTH OF VECTOR
000002  APC19KEY DS    X                    VECTOR KEY
APC19KYC EQU   X'19'                KEY IS X'19'
000003  APC19DTA DS    0X                   VECTOR DATA FIELDS
000003  APC19CSU DS    X                    COMMUNICATION STORAGE USAGE
*                                   INDICATORS
APC19NOF EQU   X'80'                NOT AN HPDT-ENABLED SESSION.
*                                  PERFORMANCE CONSTRAINTS, SHOULD
*                                  EITHER USE CSM PAGEABLE DATA
*                                  SPACE OR NON CSM STORAGE
APC19SMB EQU   X'40'                SMALLER BUFFERS RECOMMENDED FOR
*                                  CSM STORAGE USERS BECAUSE OF
*                                  RU SIZE LIMITATIONS.
*                                  NO ADDITIONAL PERFORMANCE CAN BE
*                                  GAINED USING FIXED BUFFERS.
APC19PGP EQU   X'20'                PAGEABLE BUFFERS RECOMMENDED.  HPDT
*                                  ENABLED FOR THIS SESSION.
*                                  ADDITIONAL PERFORMANCE CAN BE
*                                  GAINED USING FIXED BUFFERS.
*                                  *
000004  DS    XL2                  RESERVED
000006  APC19RU0 DS    FL4                  MAXIMUM RU SIZE OUTBOUND
000008  APC19RUI DS    FL4                  MAXIMUM RU SIZE INBOUND
*** ISTAPC1A - maps the Partner Application Capabilities vector ***
*** - Returned for these APPCCMD completions: ***
*** APPCCMD CONTROL=ALLOC ***
*** APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS ***
*** APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY ***
*** APPCCMD CONTROL=PREALLOC ***
*** APPCCMD CONTROL=RCVFMH5 ***
*** - Returned for this exit: ***
*** ATTN(CNOS) ***
*** ***
*** ...to provide partner capabilities information ***
*** for the conversation. ***
***********************************************************************
000000 ISTAPC1A DSECT MAPPING FOR PARTNER APPLICATION
* CAPABILITIES VECTOR
000000 APC1ALEN DS HL2 LENGTH OF VECTOR
000002 APC1AKEY DS X VECTOR KEY
APCIKYC EQU X'1A' KEY IS X'1A'
000003 APC1ADTA DS 0X VECTOR DATA FIELDS
000003 APC1AFL1 DS X PARTNER APPLICATION CAPABILITY
* INDICATORS
APC1APAR EQU X'C0' NEGOTIATED PARALLEL SESSION
* CAPABILITY
APC1ASSC EQU X'00' SINGLE SESSION CAPABLE
APC1ASSP EQU X'40' PENDING SINGLE STATE
APC1APSC EQU X'00' PENDING PARALLEL STATE
APC1APSC EQU X'C0' PARALLEL SESSION CAPABLE
APC1APWS EQU X'30' NEGOTIATED LEVEL OF
* PASSWORD SUBSTITUTION
APC1APSS EQU X'20' PASSWORD SUBSTITUTION
* SUPPORTED
APC1APSN EQU X'10' PASSWORD SUBSTITUTION
* NOT SUPPORTED
APC1APS U EQU X'00' PASSWORD SUBSTITUTION
* NOT SET
APC1AESS EQU X'00' PARTNER SUPPORT FOR
* EXTENDED SECURITY SENSE CODES
APC1ASSS EQU X'08' EXTENDED SECURITY SENSE CODES
* SUPPORTED
APC1ASSN EQU X'04' EXTENDED SECURITY SENSE CODES
* NOT SUPPORTED
APC1ASSU EQU X'00' EXTENDED SECURITY SENSE CODE
* NOT SET
APC1AFD EQU X'00' NEGOTIATED FDX/EXPD
* CAPABILITY
APC1AFXS EQU X'00' FDX OR HDX CONVERSATIONS AND
* EXPEDITED DATA ALLOWED
APC1AFXN EQU X'01' HDX CONVERSATIONS ONLY
APC1AFXU EQU X'00' CAPABILITY IS UNKNOWN
000004 APC1AFL2 DS X PARTNER APPLICATION CAPABILITY INDICATORS
* 
APC1ACON EQU X'C0' NEGOTIATED LEVEL OF
* SYNCHRONIZATION
APC1ACNS EQU X'00' CONFIRM, SYNC POINT AND
* BACKOUT SUPPORTED
APC1ACNN EQU X'40' CONFIRM SUPPORTED
APC1ACNU EQU X'00' SYNCHRONIZATION LEVEL NOT
* SET
APC1ASEC EQU X'20' PARTNER ACCEPTS SECURITY
* SUBFIELDS ON FMH
APC1ALV EQU X'10' PARTNER ACCEPTS REQUEST FOR
* ALREADY VERIFIED
APC1APRV EQU X'08' PARTNER ACCEPTS REQUEST FOR
* PERSISTENT VERIFICATION
* RESERVE
EQU X'07' RESERVED
000005 APC1AFL3 DS X PARTNER CHARACTERISTICS
APC1ALOC EQU X'E0' PARTNER LOCALITY STATUS
APC1AUNL EQU X'00' LOCALITY OF PARTNER UNKNOWN
APC1ARMT EQU X'00' PARTNER NOT ON SAME HOST
APC1ALCL EQU X'40' PARTNER IS ON SAME HOST SYSTEM
APC1ALUO EQU X'20' PARTNER LU SAME AS APPLICATION LU
* (LU=OWN)
*** Note: Highorder bit in vector key is on for all vectors sent from application to VTAM. ***

***********************************************************************
* ISTAPC82 - maps the XBUFLST RECEIVE vector.                      *
* - This vector is passed to VTAM on an APPCCMD CONTROL=RECEIVE    *
* when OPTCD specifies XBUFLST.                                      *
***********************************************************************

000000 ISTAPC82 DSECT                      MAPPING FOR XBUFLST RECEIVE VECTOR
000000 APC82LEN DS    HL2                  LENGTH OF VECTOR
000002 APC82KEY DS    X                    VECTOR KEY
000003 APC82KEYC EQU   X'82'                KEY IS X'82'
000003 APC82DTA DS    0X                   VECTOR DATA FIELDS
*                                    STORAGE TYPE FLAG BYTE:
*                                    ONE OR MORE OF THE FOLLOWING
APC82ECS EQU   X'80'                ECSA STORAGE REQUESTED
APC82CDS EQU   X'40'                DATA SPACE STORAGE REQUESTED
000004 APC82XBL DS    FL4                  BUFFER LENGTH (REQUIRED WHEN
*                                    IN FILL=BUFF MODE) OR ZEROS
*                                    MAXIMUM DATA TO BE RECEIVED
*                                    (OPTIONAL) OR ZEROS
000008 APC82MXD DS    FL4                  TASK TCB ADDRESS FOR CSM
*                                    STORAGE ASSOCIATION
*                                    (OPTIONAL) OR ZEROS

Application-ACB vector list (ISTVACBV)

LOC    SOURCE STATEMENT
***********************************************************************
* DATA FIELDS PASSED FROM THE APPLICATION TO VTAM.                  *
***********************************************************************
000000 ISTVACAP DSECT                     APPLID MAPPING
000000 VACAPLEN DS    X                    MAP LENGTH
000001 VACAPDTA DS    0X                   MAP DATA
* *
000000 ISTVACPW DSECT                      PASSWORD MAPPING
* *
000000 VACPWLEN DS    X                    MAP LENGTH
000001 VACPWDTA DS    0X                   MAP DATA
* *
000000 ISTVACAV DSECT                      APPLICATION VECTORLIST
*                                  POINTED TO BY ACBAPVTR
000000 VACAVLEN DS    HL2                  TOTAL LENGTH OF APPL VECTORS
000002 VACAVDTA DS    0X                   VECTOR DATA
* *
000000 ISTVACVT DSECT                      VECTOR TEMPLATE
000000 VACVTLEN DS    HL2                  VECTOR LENGTH
000002 VACVTKKEY DS    X                    VECTOR KEY
ISTVAC81 - Application Capabilities vector
- Passed to VTAM by the application at OPEN invocation for the ACB.
- Bit indicators which enable/disable application use of certain VTAM functions.

ISTVAC82 - Local Application's DCE Capability Vector
- Passed to VTAM by the application at OPEN invocation for the ACB.
- Contains the Security Mechanisms data for the Local LU.

ISTAMS01 - maps the RELEASE LEVEL vector.
- Contains identification codes for the access method product and its version, release, and modification level.
***********************************************************************

000000 ISTAMS01 DSECT RELEAS LEVEL VECTOR
000000 AMS01LEN DS X VECTOR LENGTH
000001 AMS01KEY DS X VECTOR KEY
AMS01KEY EQU X'01' KEY IS X'01'
000002 AMS01DATA DS 0CL4 VECTOR DATA
000002 AMS01PRD DS CL1 PRODUCT CODE
AMS01PRD EQU C'0' VTAM PRODUCT CODE
000003 AMS01VER DS CL1 VERSION CODE
000004 AMS01REL DS CL1 RELEASE CODE
000005 AMS02MDF DS CL1 MODIFICATION CODE
*

***********************************************************************

*** ISTAMS04 - maps the COMPONENT IDENTIFICATION vector. **
*** - This vector may be repeated. **
*** - Each component identification vector contains product **
*** identification information about a major component or **
*** feature of the VTAM licensed program. When multiple **
*** component identification vectors are present, the **
*** first one designates the base VTAM product and later **
*** vectors are features or other major VTAM components. **
*** - The vector data is in the form: C'xxxx-xxxxx-xxx'. **

***********************************************************************

000000 ISTAMS04 DSECT COMPONENT IDENTIFICATION VECTOR
000000 AMS04LEN DS X VECTOR LENGTH
000001 AMS04KEY DS X VECTOR KEY
AMS04KEY EQU X'04' KEY IS X'04'
000002 AMS04DATA DS CL14 VECTOR DATA
*

***********************************************************************

*** ISTAMS05 - maps the FUNCTION LIST vector. **
*** - The vector data is a variable-length bit string, in **
*** which each bit corresponds to a particular VTAM **
*** function. If a bit is on, the corresponding function **
*** is present in the executing release of VTAM. If a **
*** bit is off, the function is not available. If the **
*** vector is not present or if the bit string is shorter **
*** than expected, you may assume that the missing bits **
*** are zero and their corresponding functions are not **
*** available. **
*** - These indicator bits correspond to the compile-time **
*** global indicator bits in the ISTGLOBAL macro. **

***********************************************************************

000000 ISTAMS05 DSECT FUNCTION LIST VECTOR
000000 AMS05LEN DS X VECTOR LENGTH
000001 AMS05KEY DS X VECTOR KEY
AMS05KEY EQU X'05' KEY IS X'05'
000002 AMS05DTA DS 0X VECTOR DATA
000002 AMS05DT0 DS BYTE 0 OF INDICATORS
AMS05DT0 EQU X'80' NIB ENCR AND RPL CRYPT
* (CRYPTOGRAPHY)
AMS05B01 EQU X'40' ACB PARMS=NIB (COMMUNICATION
* NETWORK MANAGEMENT INTERFACE)
AMS05B02 EQU X'20' MULTIPLE-ADDRESS-SPACE
* APPLICATIONS PROGRAMS
AMS05B03 EQU X'10' AUTHORIZED PATH FOR
* COMMUNICATIONS MACROS
AMS05B04 EQU X'08' AUTHORIZED PATH FOR ALL
* RPL-BASED MACROS
AMS05B05 EQU X'04' SRBEXIT (ON APPL DEFINITION
* STATEMENT)
AMS05B06 EQU X'02' SONSECIP (ON APPL DEFINITION
* STATEMENT)
AMS05B07 EQU X'01' VTAMFRR (ON APPL DEFINITION
* STATEMENT)
*
000003 AMS05DT1 DS X BYTE 1 OF INDICATORS
AMS05B10 EQU X'80' SSCP TRACKING OF DEVICE-LU
* SESSION CAPABILITY VIA NOTIFY
* (ENABLED/DISABLED/INHIBITED)
AMS05B11 EQU X'40' RPL OPTCD=LIMPEO
AMS05B12 EQU X'20' RPL OPTCD=BUFFLST
AMS05B13 EQU X'10' RPL OPTCD=USERRH
AMS05B14 EQU X'08' ACB PARMS=USERFLD
AMS05B15 EQU X'04' RPL BRACKET=CEB
AMS05B16 EQU X'02' APPLICATION PROGRAM ASSIGNMENT OF
* SEQUENCE NUMBERS FOR EXPEDITED DFC
AMS05B17 EQU X'01' RESOURCE-IDENTIFICATION VECTOR LIST
*
000004 AMS05DT2 DS X BYTE 2 OF INDICATORS
AMS05B20 EQU X'80' ACCESS-METHOD-SUPPORT VECTOR LIST
AMS05B21 EQU X'40' RETURN OF SYSTEM RESPONSE BYTE AND

DSECTs 603
* EXTENDED RESPONSE BYTE FOR BSC 3270
* TERMINALS ATTACHED TO ACF/NCP
AMS05B22 EQU X'20'  INTRPRET
AMS05B23 EQU X'10'  VTAM API IS XRF CAPABLE
AMS05B24 EQU X'08'  SENSE ON -RSP(CINIT). CLSDST
* OPTCD=(RELEASE,SENSE)
AMS05B25 EQU X'04'  UNBIND SON CODE AND SENSE.
* CLSDST OPTCD=(RELEASE,SONCODE),
* TERMSESS OPTCD=(UNBIND,SONCODE)
AMS05B26 EQU X'02'  HOLD/RELEASE LOGON/SCIP EXIT FOR
* SESSION SETUP
AMS05B27 EQU X'01'  CINIT - NETWORK ADDRESSES IN
* VECTOR KEY X'15'

000005 AMS05DT3 DS X BYTE 3 OF INDICATORS
AMS05B30 EQU X'80'  31-BIT API
AMS05B31 EQU X'40'  NOTIFICATION OF QUEUED RESPONSES
* SUPPORTED. SEND OPTCD=(RSPQUED)
AMS05B32 EQU X'20'  APPC IS SUPPORTED
AMS05B33 EQU X'10'  ADD SUPPORT FOR USERVAR
AMS05B34 EQU X'08'  VCNS API SUPPORT FOR X.25
AMS05B35 EQU X'04'  VCNS API SUPPORT FOR TOKEN BUS,
* TOKEN RING,
AMS05B36 EQU X'02'  CROSS-MEMORY API IS SUPPORTED
AMS05B37 EQU X'01'  KEEPFRR SUPPORT (ON ACB STATEMENT)

000006 AMS05DT4 DS X BYTE 4 OF INDICATORS
AMS05B40 EQU X'80'  SRBEXIT SUPPORT (ON ACB STATEMENT)
AMS05B41 EQU X'40'  PERSISTENT LU-LU SESSIONS
AMS05B42 EQU X'20'  V.25BIS SUPPORT
AMS05B43 EQU X'10'  VTAM/NPM INTERFACE SUPPORT
AMS05B44 EQU X'08'  LU6 PLUS TRACKING SUPPORTED
AMS05B45 EQU X'04'  BYTE 4, BIT 5: RESERVED
AMS05B46 EQU X'02'  BYTE 4, BIT 6: RESERVED
AMS05B47 EQU X'01'  NETWORK QUALIFIED NAMES SUPPORTED

000007 AMS05DT5 DS X BYTE 5 OF INDICATORS
AMS05B50 EQU X'80'  MS TRANSPORT SUPPORTED
AMS05B51 EQU X'40'  PERFORMANCE MONITOR INTERFACE
* SUPPORTED
AMS05B52 EQU X'20'  QUEUED SESSION TERMINATION
* SUPPPORTED
AMS05B53 EQU X'10'  VTAM AGENT SUPPORTED
AMS05B54 EQU X'08'  GENERIC RESOURCES SUPPORTED
AMS05B55 EQU X'04'  OPTCD=KEEPSRB FOR SYNC SRB
* SUSPEND/RESUME
AMS05B56 EQU X'02'  APPLICATION VECTORS SUPPORTED ON
* ACB MACRO
AMS05B57 EQU X'01'  SETLOGON GNAMESUB SUPPORTED

000008 AMS05DT6 DS X BYTE 6 OF INDICATORS
AMS05B60 EQU X'80'  BYTE 6, BIT 0: RESERVED
AMS05B61 EQU X'40'  BYTE 6, BIT 1: RESERVED
AMS05B62 EQU X'20'  BYTE 6, BIT 2: RESERVED
AMS05B63 EQU X'10'  BYTE 6, BIT 3: RESERVED
AMS05B64 EQU X'08'  BYTE 6, BIT 4: RESERVED
AMS05B65 EQU X'04'  BYTE 6, BIT 5: RESERVED
AMS05B66 EQU X'02'  BYTE 6, BIT 6: RESERVED
AMS05B67 EQU X'01'  BYTE 6, BIT 7: RESERVED

000009 AMS05DT7 DS X BYTE 7 OF INDICATORS
AMS05B70 EQU X'80'  BYTE 7, BIT 0: RESERVED
AMS05B71 EQU X'40'  BYTE 7, BIT 1: RESERVED
AMS05B72 EQU X'20'  BYTE 7, BIT 2: RESERVED
AMS05B73 EQU X'10'  BYTE 7, BIT 3: RESERVED
AMS05B74 EQU X'08'  BYTE 7, BIT 4: RESERVED
AMS05B75 EQU X'04'  BYTE 7, BIT 5: RESERVED
AMS05B76 EQU X'02'  BYTE 7, BIT 6: RESERVED
AMS05B77 EQU X'01'  BYTE 7, BIT 7: RESERVED

***********************************************************************
*** ISTAMS06 - maps the LU6.2 SUPPORT FUNCTION LIST vector.            **
*** - The vector data is a variable-length string of byte              **
*** indicators, each byte corresponding to a particular              **
*** LU6.2 function. Each byte will have a value showing              **
*** that its corresponding function is either supported,             **
*** or not supported, or supported on a "pass-through" basis.**       **
*** (Pass-through functions are those which VTAM does not **           **
*** directly provide but provides the ability for the              **
*** application to create the support itself.)                     **
*** If the vector is not present or if the byte string               **

***********************************************************************
*** is shorter than expected, you may assume that the **
*** missing bytes are zero and their corresponding **
*** functions are not supported. **
*** - These indicator bytes correspond to the compile-time **
*** global indicators in the ISTGAPPC macro.
**********************************************************************

000000 ISTAMS06 DSECT                      LU6.2 SUPPORT FUNCTION LIST VECTOR
000000 ISTAMS06 LEN DS  X                    VECTOR LENGTH
000000 ISTAMS06 KEY DS  X                    VECTOR KEY
000000 ISTAMS06 KEYC EQU X'06'              KEY IS X'06'
000000 ISTAMS06 DATA DS  6X                  VECTOR DATA
000000 ISTAMS06 D01 DS  X                    01. CONVERSATIONS BETWEEN TPS
*                                           AT SAME LU
*000000 ISTAMS06 D02 DS  X                    02. DELAYED SESSION
*                                           ALLOCATION
*000000 ISTAMS06 D03 DS  X                    03. IMMEDIATE SESSION
*                                           ALLOCATION
*000000 ISTAMS06 D04 DS  X                    04. SYNC POINT SERVICES
*000000 ISTAMS06 D05 DS  X                    05. PROGRAM RECONNECT
*000000 ISTAMS06 D06 DS  X                    06. RESERVED
*000000 ISTAMS06 D07 DS  X                    07. SESSION-LEVEL LU-LU
*                                           VERIFICATION
*000000 ISTAMS06 D08 DS  X                    08. USERID VERIFICATION
*000000 ISTAMS06 D09 DS  X                    09. PROGRAM SUPPLIED USERID
*                                           AND PASSWORD
*000000 ISTAMS06 D10 DS  X                    10. USERID AUTHORIZATION
*000000 ISTAMS06 D11 DS  X                    11. PROFILE VERIFICATION AND
*                                           AUTHORIZATION
*000000 ISTAMS06 D12 DS  X                    12. RESERVED
*000000 ISTAMS06 D13 DS  X                    13. PROFILE PASSTHROUGH
*000000 ISTAMS06 D14 DS  X                    14. PROGRAM-SUPPLIED PROFILE
*000000 ISTAMS06 D15 DS  X                    15. SEND PERSISTENT
*                                           VERIFICATION
*000000 ISTAMS06 D16 DS  X                    16. RECEIVE PERSISTENT
*                                           VERIFICATION
*000000 ISTAMS06 D17 DS  X                    17. PIP DATA
*000000 ISTAMS06 D18 DS  X                    18. LOGGING OF DATA IN SYSTEM
*                                           LOG
*000000 ISTAMS06 D19 DS  X                    19. FLUSH LU SEND BUFFER
*000000 ISTAMS06 D20 DS  X                    20. LUW IDENTIFIER
*000000 ISTAMS06 D21 DS  X                    21. PREPARE TO RECEIVE
*000000 ISTAMS06 D22 DS  X                    22. LONG LOCKS
*000000 ISTAMS06 D23 DS  X                    23. POST ON RECEIPT WITH WAIT
*000000 ISTAMS06 D24 DS  X                    24. POST ON RECEIPT WITH TEST
*                                           FOR POSTING
*000000 ISTAMS06 D25 DS  X                    25. RECEIVE-IMMEDIATE
*000000 ISTAMS06 D26 DS  X                    26. TEST FOR REQUEST-TO-SEND
*                                           RECEIVED
*000000 ISTAMS06 D27 DS  X                    27. DATA MAPPING
*000000 ISTAMS06 D28 DS  X                    28. FMH APPLICATION-DATA
*000000 ISTAMS06 D29 DS  X                    29. GET ATTRIBUTES
*000000 ISTAMS06 D30 DS  X                    30. GET CONVERSATION-TYPE
*000000 ISTAMS06 D31 DS  X                    31. MAPPED CONVERSATION LU
*                                           SERVICES COMPONENT
*000000 ISTAMS06 D32 DS  X                    32. CHANGE_SESSION_LIMIT VERB
*000000 ISTAMS06 D33 DS  X                    33. MIN_CONWINNERS_TARGET
*                                           PARAMETER
*000000 ISTAMS06 D34 DS  X                    34. RESPONSIBLE(TARGET)
*                                           PARAMETER
*000000 ISTAMS06 D35 DS  X                    35. DRAIN_TARGET(NO) PARAMETER
*000000 ISTAMS06 D36 DS  X                    36. FORCE PARAMETER
*000000 ISTAMS06 D37 DS  X                    37. ACTIVATE_SESSION VERB
*000000 ISTAMS06 D38 DS  X                    38. DEACTIVATE_SESSION VERB
*000000 ISTAMS06 D39 DS  X                    39. LU-PARAMETER VERBS
*000000 ISTAMS06 D40 DS  X                    40. LU-LU SESSION LIMIT
*000000 ISTAMS06 D41 DS  X                    41. LOCALLY-KNOWN LU NAMES
*000000 ISTAMS06 D42 DS  X                    42. UNINTERPRETED LU NAMES
*000000 ISTAMS06 D43 DS  X                    43. SINGLE-SESSION
**                                        RE-INITIATION
*000000 ISTAMS06 D44 DS  X                    44. ALTERNATE CODE PROCESSING
*000000 ISTAMS06 D45 DS  X                    45. MAXIMUM RU SIZE BOUNDS
*000000 ISTAMS06 D46 DS  X                    46. SESSION-LEVEL MANDATORY
*                                           CRYPTOGRAPHY
*000000 ISTAMS06 D47 DS  X                    47. CONTENTION WINNER
*                                           AUTOMATIC ACTIVATION LIMIT
*000000 ISTAMS06 D48 DS  X                    48. CONWINNER SESSION
*                                           ALLOCATION
*000000 ISTAMS06 D49 DS  X                    49. ENHANCED SECURITY (SAME)
*000000 ISTAMS06 D50 DS  X                    50. SESSION-LEVEL SELECTIVE
*                                           CRYPTOGRAPHY
*000000 ISTAMS06 D51 DS  X                    51. CONVERSATION GROUP SUPPORT
*000000 ISTAMS06 D52 DS  X                    52. ALLOCATE WHEN SESSION FREE

DSECTs 605
Resource-information vector list (ISTRIVL)

Loc Source Statement
000000 ISTRIVL DSECT MAPPING FOR RESOURCE INFORMATION
   VECTOR LIST POINTED TO BY ACBRIVL
000000 RIVLLEN DS HL2 TOTAL LENGTH OF VECTORS
000002 RIVLDATA DS 0X VECTOR DATA
   ***********************************************************************
   ***  GENERALIZED MAPPING FOR EXAMINING COMMON FIELDS IN ALL ACB      
   ***  VECTORS IN THE VECTOR LIST POINTED TO BY ACBRIVL             
   *** **********************************************************************
000000 ISTRIVVT DSECT VECTOR TEMPLATE @Y3A
000000 RIVVTLEN DS X VECTOR LENGTH @Y3A
000001 RIVVTKYX DS X VECTOR KEY @Y3A
000002 RIVVTDATA DS 0X VECTOR DATA @Y3A
   ***********************************************************************
   *** ISTRIV02 - maps the application's network name vector.           
   ***          - The name is specified by the name field of the         
   ***            application definition statement.                     
   ***          - This is obtained from the NAME ON APPL STATEMENT.      
   *** ***********************************************************************
000000 ISTRIV02 DSECT APPLICATION NETWORK NAME VECTOR
   (FROM NAME ON APPL STATEMENT)
000000 RIV02LEN DS X VECTOR LENGTH
000001 RIV02KEY DS X VECTOR KEY
    RIV02KYC EQU X'02' KEY IS X'02'
000002 RIV02DTA DS CL8 VECTOR DATA
   ***********************************************************************
   *** ISTRIV03 - maps the application’s ACB name vector.               
   ***          - This is supplied by the APPLID operand on the ACB     
   ***            statement or can be supplied by the operating         
   ***            system. During OPEN ACB, VTAM will search for the      
   ***            application’s characteristics by matching the ACB     
   ***            APPLID value to an RDTE with the application’s        
   ***            ACBNAME. If ACBNAME was not coded for the             
   ***            application, VTAM will search for a match with an      
   ***            RDTE containing the application’s network name.       
   ***          - This is obtained from the APPLID on ACB MACRO.        
   *** ***********************************************************************
000000 ISTRIV03 DSECT APPLICATION ACB NAME VECTOR
   (FROM APPLID ON ACB MACRO)
000000 RIV03LEN DS X VECTOR LENGTH
000001 RIV03KEY DS X VECTOR KEY
    RIV03KYC EQU X'03' KEY IS X'03'
000002 RIV03DTA DS CL8 VECTOR DATA
   ***********************************************************************
   *** ISTRIV06 - maps the network name in which the host resides.      
   ***          - This is obtained from the NETID START OPTION.          
   ***          - If NETID start option is not specified, this value     
   ***            will be blanks.                                       
   *** ***********************************************************************
000000 ISTRIV06 DSECT NETWORK NAME VECTOR
   (FROM NETID START OPTION)
000000 RIV06LEN DS X VECTOR LENGTH
000001 RIV06KEY DS X VECTOR KEY
    RIV06KYE EQU X'06' KEY IS X'06'
000002 RIV06DTA DS CL8 VECTOR DATA
   ***********************************************************************
   *** ISTRIV07 - maps the SSCP Name vector.                            
   ***          - This is obtained from the SSCPNAME START OPTION        

000036 AMS06D53 DS X 53. FULL-DUPLEX
000037 AMS06D54 DS X 54. APPCCMD VECTOR LISTS
000038 AMS06D55 DS X 55. QUEUED RCVFMH5
000039 AMS06D56 DS X 56. HIGH PERFORMANCE DATA
  * - This is obtained from the SSCPNAME START OPTION
00003A AMS06D57 DS X 57. APPCCMD SENDRCV
00003B AMS06D58 DS X 58. INTRA-LU CONVERSATIONS
00003C AMS06D59 DS X 59. PASSWORD SUBSTITUTION
00003D AMS06D60 DS X 60. EXTENDED SECURITY SENSE
00003E AMS06D61 DS X 61. DCE SECURITY SERVICES

606 z/OS Communications Server: SNA Programmer's LU 6.2 Reference
***********************************************************************
000000 ISTRIV07 DSECT                      SSCP NAME VECTOR
*                                   (FROM SSCPNAME START OPTION)
000000 RIV07LEN DS    X                    VECTOR LENGTH
000001 RIV07KEY DS    X                    VECTOR KEY
RIV07KYC EQU   X'07'                KEY IS X'07'
000002 RIV07DTA DS    CL8                  VECTOR DATA
*                                   (DEFAULT IS 'VTAM')
*
***********************************************************************
*** ISTRIV08 - maps the Host Subarea PU Network Name vector.         **
***          - This is obtained from the HOSTPU START OPTION         **
***            If HOSTPU start option is not specified, the name     **
***            will default to 'ISTPUS '.                           **
***********************************************************************
000000 ISTRIV08 DSECT                      HOST SUBAREA PU NETWORK NAME VECTOR
*                                   (FROM HOSTPU START OPTION)
000000 RIV08LEN DS    X                    VECTOR LENGTH
000001 RIV08KEY DS    X                    VECTOR KEY
RIV08KYC EQU   X'08'                KEY IS X'08'
000002 RIV08DTA DS    CL8                  VECTOR DATA
*                                   (DEFAULT IS 'ISTPUS')
*
***********************************************************************
*** ISTRIV09 - maps the Host Subarea PU network address vector.      **
***          - It contains the network address of the host           **
***            subarea PU.                                           **
***********************************************************************
000000 ISTRIV09 DSECT                      HOST SUBAREA PU NETWORK ADDRESS
*                                   (DEFAULT IS 'ISTPUS')
000000 RIV09LEN DS    X                    VECTOR LENGTH
000001 RIV09KEY DS    X                    VECTOR KEY
RIV09KYC EQU   X'09'                KEY IS X'09'
000002 RIV09DTA DS    XL6                  VECTOR DATA
*
***********************************************************************
*** ISTRIV0A - maps the maximum subarea vector.                      **
***          - Contains the maximum subarea number that is valid     **
***            for the host's domain.                                **
***            This is obtained from the MAXSUBA START OPTION        **
***********************************************************************
000000 ISTRIV0A DSECT                      MAXIMUM SUBAREA NUMBER VECTOR
*                                   (FROM MAXSUBA START OPTION)
000000 RIV0ALEN DS    X                    VECTOR LENGTH
000001 RIV0AKEY DS    X                    VECTOR KEY
RIV0AKYC EQU   X'0A'                KEY IS X'0A'
000002 RIV0ADTA DS    X                    VECTOR DATA
*
***********************************************************************
*** ISTRIV0B - maps the LU 6.2 application definition vector.        **
***            After the LU 6.2 application program has issued an     **
***            open ACB, the LU 6.2 application program may use      **
***            this vector to determine the values coded on the      **
***            APPL definition statement.                            **
***          - This is obtained from the APPL STATEMENT PARAMETERS   **
***********************************************************************
000000 ISTRIV0B DSECT                      LU 6.2 APPL DEFINITION VECTOR
*                                   (FROM APPL STATEMENT PARAMETERS)
000000 RIV0BLEN DS    X                    VECTOR LENGTH
000001 RIV0BKEY DS    X                    VECTOR KEY
RIV0BKYC EQU   X'0B'                KEY IS X'0B'
000002 RIV0BDTA DS    0X                   VECTOR DATA
000002          DS    X                    RESERVED
RIV0BSLV EQU   X'C0'                SESSION-LEVEL LU-LU VERIFICATION
*                                   BIT MASK
RIV0BLSR EQU   X'80'                REQUIRED
RIV0BLSL EQU   X'40'                OPTIONAL
RIV0BSLN EQU   X'00'                NONE
000003 RIV0BCLS D5    X                    CONVERSATION SECURITY ACCEPTANCE
RIV0BCLN EQU   X'01'                NONE
RIV0BCLC EQU   X'02'                CONV
RIV0BCLA EQU   X'03'                ALREADYV
RIV0BCLP EQU   X'04'                PERSISTV
RIV0BCLV EQU   X'05'                AVPV
000004 RIV0BLFG DS    X                    MISCELLANEOUS FLAGS
RIV0BBDL EQU   X'80'                DDRAINL=ALLOW
RIV0BDRL EQU   X'40'                DRESPL=ALLOW
RIV0BATA EQU   X'20'                ATNCLSS=ALLOW
RIV0BSYP EQU   X'10'                SYNCLVL=SYNCPT
RIV0BOPC EQU   X'08'                OPERCNOS=ALLOW
000005 DS    X                    RESERVED
000006 RIV0BDSL DS    HL2                  DSESLSIN VALUE
***********************************************************************
*** ISTRIV0C - maps the common application definition vector. **
*** After the application program has issued an open for **
*** its ACB, the application may examine this vector to **
*** determine the values coded on the APPL definition **
*** statement for common application definition keywords. **
*** - This is obtained from the APPL STATEMENT PARAMETERS **
**********************************************************************

***          - This is obtained from the APPL STATEMENT PARAMETERS   **
***            statement for common application defination keywords. **
***            determine the values coded on the APPL definition     **
***            After the application program has issued an open for  **
***            the ACB, the application may examine this vector to   **
***            examine this vector to **
*** ISTRIV0C - maps the common application definition vector. **
**********************************************************************

*** ISTRIV11 - maps the APPCCMD vector area length vector. **
*** - It contains the absolute minimum length and the **
*** recommended minimum length for full use of the **
*** APPCCMD vector area. **
**********************************************************************

*** ISTRIV12 - maps the application to VTAM vector keys vector. **
*** - It contains a list of all ACB vector keys that **
*** VTAM will process. Constants for the ACB vectors are **
*** located in ISTVACBVY. **
**********************************************************************
*** ISTRIV13 - maps the Performance Monitor vector. **
*** Identifies a table of Performance Data vector fields **
*** (within ISTXPL) that have been retired by the **
*** Performance Monitor Interface since its inception. **
***********************************************************************

ISTRIV13 DSECT PERFORMANCE MONITOR VECTOR @L1A

RIV13LEN DS X VECTOR LENGTH @L1A
RIV13KEY DS X VECTOR KEY @L1A
RIV13KEYC EQU X'13' KEY IS X'13' @L1A
RIV13ENT DS HL2 NUMBER OF ENTRIES IN TABLE @L2A
RIV13RFT DS AL4 RETIRED FIELDS TABLE ADDRESS @L2A
RIV13ELN DS HL2 LENGTH OF EACH ENTRY @L2A

RIV13TBL DSECT RETIRED FIELDS TABLE ENTRY @L1A

RIV13VID DS 0CL6 ID OF AFFECTED VECTOR @L1A
RIV13MAJ DS CL2 MAJOR CATEGORY @L1A
RIV13SUB DS CL2 SUBCATEGORY @L1A
RIV13REC DS CL2 RECORD TYPE @L1A
RIV13FLD DS 0CL4 FIELD POSITION WITHIN VECTOR @L1A
RIV13OFF DS HL2 FIELD OFFSET @L1A
RIV13FLG DS BL1 FLAG BYTE @L1A
RIV13BIT EQU X'01' DATA TYPE INDICATOR
RIV13LNG DS XL1 FIELD LENGTH IF NOT BITSTRING,
* FOR BITSTRING FIELD

Extended buffer list entry (ISTBLXEN)

LOC    SOURCE STATEMENT
000000 ISTBLXEN DSECT
000000 BLXEN_CSM DS 0CL28 THIS AREA MAPS THE CSM
*    BUFFER DESCRIPTOR
000000 BS X X RESERVED. THIS FIELD MUST BE
*    SET TO ZERO.
000001 BLXEN_SOURCE DS X BUFFER SOURCE
*    BLXEN_CECSA EQU X'80' INDICATES THAT THE STORAGE
*    REFERENCED IN THE LIST IS
*    CSM ECSA
*    BLXEN_CDSPACE EQU X'40' INDICATES THAT THE STORAGE
*    REFERENCED IN THE LIST IS
*    CSM DATA SPACE
000002 BLXEN_TYPE DS X BUFFER TYPE
*    BLXEN_FIXED EQU X'80' INDICATES THAT THE STORAGE IS
*    IN A GUARANTEED TO BE FIXED
*    STATE
*    BLXEN_PAGEABLE EQU X'40' INDICATES THAT THE STORAGE IS
*    IN A GUARANTEED TO BE PAGEABLE
*    STATE
*    BLXEN_PAGEELIG EQU X'20' INDICATES THAT THE STORAGE
*    IS ELIGIBLE TO BE PAGEFREED BY
*    CSM
000003 BS XL1 RESERVED
000004 BLXEN_CTKN DS XL12 CSM TOKEN
000010 BLXEN_ALET DS F CSM DATA SPACE ALET
000014 BLXEN_AREA DS A POINTER TO DATA
000006 BLXEN_RLEN DS F LENGTH OF DATA
00001C BLXEN_RLENA DS F LENGTH OF DATA ACCEPTED BY
*    VTAM ON A REQUEST TO SEND
*    DATA.
*    THIS FIELD SHOULD BE SET
*    TO ZERO BY THE APPLICATION.
*    VTAM SETS THIS FIELD
*    TO REFLECT THE AMOUNT OF
*    DATA REFERENCED BY XBUFLST
*    THAT HAS BEEN ACCEPTED BY
<table>
<thead>
<tr>
<th>000020</th>
<th>BLXEN_VAFLAGS</th>
<th>DS X</th>
<th>VTAM and APPL FLAGS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLXEN_OWNACC</td>
<td>EQU X'80'</td>
<td>VTAM HAS ACCEPTED OWNERSHIP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OF THE CSM BUFFER</td>
</tr>
<tr>
<td>000021</td>
<td></td>
<td>DS XL15</td>
<td>RESERVED</td>
</tr>
</tbody>
</table>
Chapter 4. Summary of register usage

Table 3 on page 611 shows what VTAM does with the general-purpose registers before it returns control to the application program at the next sequential instruction. It indicates which registers are left unchanged by the VTAM macroinstructions and which ones can be modified between the time the macroinstruction is executed and control is returned to the application program. The table also shows the disposition of the registers when any of the exit routines receive control. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for further details on how to handle macroinstruction errors.

Table 3. Register contents upon return of control

<table>
<thead>
<tr>
<th>Register 0</th>
<th>Register 1</th>
<th>Register 2-12</th>
<th>Register 13</th>
<th>Register 14</th>
<th>Register 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upon return from OPEN and CLOSE macroinstructions</td>
<td>Unpredictable</td>
<td>Unpredictable</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>Return code</td>
</tr>
<tr>
<td>Upon return from RPL-based macroinstructions, including CHECK</td>
<td>See footnote 2</td>
<td>Address of RPL</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>Upon return from GENCB</td>
<td>Error return code or control block address</td>
<td>Control block address</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>Upon return from SHOWCB, MODCB, or TESTCB</td>
<td>Error return code</td>
<td>Unpredictable</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>Upon invocation of LERAD or SYNAD exit routines</td>
<td>Recovery action return code</td>
<td>Address of RPL</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>Return address</td>
</tr>
</tbody>
</table>

1 Register 13 must indicate the address of an 18-word save area when the macroinstruction is executed.
2 If the operation completed normally, register 15 is set to 0. For some macroinstructions completing normally but with a special condition, register 0 is also set. If an error occurred and the LERAD or SYNAD exit routine has been invoked, registers 0 and 15 contain the values set in them by the exit routine. If an error occurred and no LERAD or SYNAD exit routine exists, VTAM sets register 15 to 4 and places a recovery action return code in register 0 (if the error is that the ACB is not open, register 15 is set to decimal 32 and the RPL request code is set in register 0).
3 When GENCB completes successfully (register 15 is set to 0), register 1 contains the address of the generated control blocks and register 0 contains the length of the control blocks, in bytes.
4 If GENCB, SHOWCB, MODCB, or TESTCB completes unsuccessfully (with register 15 not set to 0), register 1 is unpredictable and register 0 contains an error code (if register 15 is set to 4 or 12) or else is unpredictable.
### Table 3. Register contents upon return of control (continued)

<table>
<thead>
<tr>
<th>Register 0</th>
<th>Register 1</th>
<th>Register 2-12</th>
<th>Register 13</th>
<th>Register 14</th>
<th>Register 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upon invocation of other EXLST exit routines</td>
<td>Unpredictable</td>
<td>Address of VTAM-supplied parameter list</td>
<td>Unpredictable</td>
<td>Return address</td>
<td>Address of exit routine</td>
</tr>
<tr>
<td>Upon invocation of RPL-based exit routines</td>
<td>Unpredictable</td>
<td>Address of RPL</td>
<td>Unpredictable</td>
<td>Return address</td>
<td>Address of exit routine</td>
</tr>
</tbody>
</table>
Appendix A. Architectural specifications

This appendix lists documents that provide architectural specifications for the SNA Protocol.

The APPN Implementers' Workshop (AIW) architecture documentation includes the following architectural specifications for SNA APPN and HPR:

- APPN Architecture Reference (SG30-3422-04)
- APPN Branch Extender Architecture Reference Version 1.1
- APPN Dependent LU Requester Architecture Reference Version 1.5
- APPN Extended Border Node Architecture Reference Version 1.0
- APPN High Performance Routing Architecture Reference Version 4.0
- SNA Formats (GA27-3136-20)
- SNA Technical Overview (GC30-3073-04)

For more information, see the AIW documentation page at http://www.ibm.com/support/docview.wss?rs=852&uid=swg27017843.

The following RFC also contains SNA architectural specifications:

- RFC 2353 APPN/HPR in IP Networks APPN Implementers' Workshop Closed Pages Document

Appendix B. Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you can view the information through the z/OS Internet Library website http://www.ibm.com/systems/z/os/zos/library/bkserw/ or IBM Knowledge Center http://www.ibm.com/support/knowledgecenter/. If you continue to experience problems, send a message to Contact z/OS web page (www.ibm.com/systems/z/os/zos/webqs.html) or write to:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

• Use assistive technologies such as screen readers and screen magnifier software
• Operate specific or equivalent features using only the keyboard
• Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. See z/OS TSO/E Primer, z/OS TSO/E User’s Guide, and z/OS ISPF User’s Guide Vol I for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

One exception is command syntax that is published in railroad track format, which is accessible using screen readers with IBM Knowledge Center, as described in #accessibility/ddsd.

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing IBM Knowledge Center using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.
Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 \* FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* \* FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should see separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- A question mark (?) means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

- An exclamation mark (!) means a default syntax element. A dotted decimal number followed by the! symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP applies only to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

- An asterisk (*) means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.

3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.
   • + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.
This information was developed for products and services that are offered in the USA or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 United States of America

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing Legal and Intellectual Property Law IBM Japan Ltd. 19-21, Nihonbashi-Hakozakicho, Chuo-ku Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for the Knowledge Centers. Use of hyperlinks in other output formats of this information is at your own risk.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation Site Counsel 2455 South Road Poughkeepsie, NY 12601-5400 USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.
Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM’s future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

**Applicability**

These terms and conditions are in addition to any terms of use for the IBM website.

**Personal use**

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

**Commercial use**

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

**Rights**

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.
IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

**IBM Online Privacy Statement**

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user’s name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM’s Privacy Policy at ibm.com/privacy and IBM’s Online Privacy Statement at ibm.com/privacy/details in the section entitled “Cookies, Web Beacons and Other Technologies,” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at ibm.com/software/info/product-privacy.

**Policy for unsupported hardware**

Various z/OS elements, such as DFSMS, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

**Minimum supported hardware**

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: [IBM Lifecycle Support for z/OS](www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.
Programming interface information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of z/OS Communications Server.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.
This bibliography contains descriptions of the documents in the z/OS Communications Server library.

z/OS Communications Server documentation is available online at the z/OS Internet Library web page at http://www.ibm.com/systems/z/os/zos/library/bkserv/.

z/OS Communications Server library updates

Updates to documents are also available on RETAIN and in information APARs (info APARs). Go to http://www.software.ibm.com/support to view information APARs.

• z/OS V2R1 Communications Server New Function APAR Summary
• z/OS V2R2 Communications Server New Function APAR Summary
• z/OS V2R3 Communications Server New Function APAR Summary

z/OS Communications Server information

z/OS Communications Server product information is grouped by task in the following tables.

### Planning

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: New Function Summary</td>
<td>GC27-3664</td>
<td>This document is intended to help you plan for new IP or SNA functions, whether you are migrating from a previous version or installing z/OS for the first time. It summarizes what is new in the release and identifies the suggested and required modifications needed to use the enhanced functions.</td>
</tr>
<tr>
<td>z/OS Communications Server: IPv6 Network and Application Design Guide</td>
<td>SC27-3663</td>
<td>This document is a high-level introduction to IPv6. It describes concepts of z/OS Communications Server's support of IPv6, coexistence with IPv4, and migration issues.</td>
</tr>
</tbody>
</table>

### Resource definition, configuration, and tuning

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: IP Configuration Guide</td>
<td>SC27-3650</td>
<td>This document describes the major concepts involved in understanding and configuring an IP network. Familiarity with the z/OS operating system, IP protocols, z/OS UNIX System Services, and IBM Time Sharing Option (TSO) is recommended. Use this document with the z/OS Communications Server: IP Configuration Reference.</td>
</tr>
</tbody>
</table>
### Title

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
</table>
| z/OS Communications Server: IP Configuration Reference     | SC27-3651 | This document presents information for people who want to administer and maintain IP. Use this document with the z/OS Communications Server: IP Configuration Guide. The information in this document includes:  
- TCP/IP configuration data sets  
- Configuration statements  
- Translation tables  
- Protocol number and port assignments |
# Customization

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
</table>
| z/OS Communications Server: SNA Customization | SC27-3666 | This document enables you to customize SNA, and includes the following information:  
- Communication network management (CNM) routing table  
- Logon-interpret routine requirements  
- Logon manager installation-wide exit routine for the CLU search exit  
- TSO/SNA installation-wide exit routines  
- SNA installation-wide exit routines |

# Writing application programs

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: IP Sockets Application Programming Interface Guide and Reference</td>
<td>SC27-3660</td>
<td>This document describes the syntax and semantics of program source code necessary to write your own application programming interface (API) into TCP/IP. You can use this interface as the communication base for writing your own client or server application. You can also use this document to adapt your existing applications to communicate with each other using sockets over TCP/IP.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP CICS Sockets Guide</td>
<td>SC27-3649</td>
<td>This document is for programmers who want to set up, write application programs for, and diagnose problems with the socket interface for CICS® using z/OS TCP/IP.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP IMS Sockets Guide</td>
<td>SC27-3653</td>
<td>This document is for programmers who want application programs that use the IMS TCP/IP application development services provided by the TCP/IP Services of IBM.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Programmer’s Guide and Reference</td>
<td>SC27-3659</td>
<td>This document describes the syntax and semantics of a set of high-level application functions that you can use to program your own applications in a TCP/IP environment. These functions provide support for application facilities, such as user authentication, distributed databases, distributed processing, network management, and device sharing. Familiarity with the z/OS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Programming</td>
<td>SC27-3674</td>
<td>This document describes how to use SNA macroinstructions to send data to and receive data from (1) a terminal in either the same or a different domain, or (2) another application program in either the same or a different domain.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Programmer’s LU 6.2 Guide</td>
<td>SC27-3669</td>
<td>This document describes how to use the SNA LU 6.2 application programming interface for host application programs. This document applies to programs that use only LU 6.2 sessions or that use LU 6.2 sessions along with other session types. (Only LU 6.2 sessions are covered in this document.)</td>
</tr>
<tr>
<td>Title</td>
<td>Number</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Programmer's LU 6.2 Reference</td>
<td>SC27-3670</td>
<td>This document provides reference material for the SNA LU 6.2 programming interface for host application programs.</td>
</tr>
<tr>
<td>z/OS Communications Server: CSM Guide</td>
<td>SC27-3647</td>
<td>This document describes how applications use the communications storage manager.</td>
</tr>
</tbody>
</table>

## Diagnosis

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: IP Diagnosis Guide</td>
<td>GC27-3652</td>
<td>This document explains how to diagnose TCP/IP problems and how to determine whether a specific problem is in the TCP/IP product code. It explains how to gather information for and describe problems to the IBM Software Support Center.</td>
</tr>
<tr>
<td>z/OS Communications Server: ACF/TAP Trace Analysis Handbook</td>
<td>GC27-3645</td>
<td>This document explains how to gather the trace data that is collected and stored in the host processor. It also explains how to use the Advanced Communications Function/Trace Analysis Program (ACF/TAP) service aid to produce reports for analyzing the trace data information.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures and z/OS Communications Server: SNA Diagnosis Vol 2, FFST Dumps and the VIT</td>
<td>GC27-3667 GC27-3668</td>
<td>These documents help you identify an SNA problem, classify it, and collect information about it before you call the IBM Support Center. The information collected includes traces, dumps, and other problem documentation.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Data Areas Volume 1 and z/OS Communications Server: SNA Data Areas Volume 2</td>
<td>GC31-6852 GC31-6853</td>
<td>These documents describe SNA data areas and can be used to read an SNA dump. They are intended for IBM programming service representatives and customer personnel who are diagnosing problems with SNA.</td>
</tr>
</tbody>
</table>

## Messages and codes

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
</table>
| z/OS Communications Server: SNA Messages | SC27-3671 | This document describes the ELM, IKT, IST, IUT, IVT, and USS messages. Other information in this document includes:  
• Command and RU types in SNA messages  
• Node and ID types in SNA messages  
• Supplemental message-related information |
<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: IP Messages Volume 1 (EZA)</td>
<td>SC27-3654</td>
<td>This volume contains TCP/IP messages beginning with EZA.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Messages Volume 2 (EZB, EZD)</td>
<td>SC27-3655</td>
<td>This volume contains TCP/IP messages beginning with EZB or EZD.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Messages Volume 3 (EZY)</td>
<td>SC27-3656</td>
<td>This volume contains TCP/IP messages beginning with EZY.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Messages Volume 4 (EZZ, SNM)</td>
<td>SC27-3657</td>
<td>This volume contains TCP/IP messages beginning with EZZ and SNM.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP and SNA Codes</td>
<td>SC27-3648</td>
<td>This document describes codes and other information that appear in z/OS Communications Server messages.</td>
</tr>
</tbody>
</table>
Index

A

AAREA keyword
accepting a session 134
allocation request for specific session 19
allocation request, conditional, without wait 51
allocation requests for contention-winner session 30
allocation requests that can be queued 9
as pointer to RPL extension 359
CNOS requests 140
conditional deallocation 90
conditional deallocation that includes data 98
deallocation due to program errors 63
deallocation due to service errors 69
deallocation due to timer errors 76
deallocation due to user-specified criteria 83
defining LU-mode table values 152
displaying LU-mode table values 158
entering receive state conditionally 220
entering receive state conditionally and including data 228
entering receive state unconditionally 249
entering receive state unconditionally and including data 239
flushing the buffer 443
immediate allocation request 41
indicating the beginning of a synchronization exchange 492
indicating the end of a synchronization exchange 496
obtain status on information available on a specified conversation 516
obtain status on information from any active conversation 501
obtain status on information immediately available from any active conversation 505
queued deallocation due to program, service, timer, or user errors 127
receive normal information immediately available from any conversation 325
receive normal information immediately available on a specific conversation 336
receiving an FMH-5 297
receiving expedited data from the specified conversation 278
receiving expedited data immediately available in any-mode 263
receiving expedited data immediately from the specified conversation 270
receiving expedited data in any-mode 256
rejecting a session 148
releasing a suspended session 483
replying to a confirmation request 389
requesting permission to enter send state 450
resetting the continuation mode 375
AAREA keyword (continued)
restoring a mode 164
sending an error notification 430
sending data 396
sending data and flushing the buffer 418
sending data conditionally 381, 408
sending expedited data on a full-duplex session 457
specific-mode RECEIVES 348
suspending a subsequent conversation 487
terminating a session 359
terminating a session due to errors 369
terminating session due to protocol error or cleanup deactivation 364
unconditional deallocation 119
unconditional deallocation that includes data 109
abnormal conversation deallocation
LU Services errors (ABNDSERV) 67, 124
reference material (macroinstruction syntax and operands) 60, 81
timing errors (ABNDTIME) 74, 124
transaction program errors (ABNDPROG) 60, 124
user-defined errors (ABNDUSER) 81, 124
ACB keyword
accepting a session 134
allocation request for contention-winner session 30
allocation request for specific session 20
allocation request, conditional, without wait 51
allocation requests that can be queued 9
any-mode RECEIVES 314
CNOS requests 140
conditional deallocation 90
conditional deallocation that includes data 98
deallocation due to program errors 63
deallocation due to service errors 69
deallocation due to timer errors 76
deallocation due to user-specified criteria 83
defining LU-mode table values 152
displaying LU-mode table values 158
entering receive state conditionally 220
entering receive state conditionally and including data 228
entering receive state unconditionally 249
entering receive state unconditionally and including data 239
flushing the buffer 443
immediate allocation request 41
indicating the beginning of a synchronization exchange 492
indicating the end of a synchronization exchange 496
obtain status on information available on a specified conversation 516
obtain status on information from any active conversation 501
obtain status on information immediately available from any active conversation 505
obtain status on information immediately available on a specified conversation 510
queued deallocation due to program, service, timer, or user errors 127
receive normal information immediately available from any conversation 325
receive normal information immediately available on a specific conversation 336
receiving an FMH-5 297
receiving expedited data from the specified conversation 278
receiving expedited data immediately available in any-mode 263
receiving expedited data immediately from the specified conversation 270
receiving expedited data in any-mode 256
rejecting a session 148
releasing a suspended session 483
replying to a confirmation request 389
requesting permission to enter send state 450
resetting the continuation mode 375
ACB keyword (continued)
queued deallocation due to program, service, timer, or user errors 127
receive normal information immediately available from any conversation 325
receive normal information immediately available on a specific conversation 336
receiving an FMH-5 297
receiving expedited data from the specified conversation 278
receiving expedited data immediately available in any-mode 263
receiving expedited data immediately from the specified conversation 270
receiving expedited data in any-mode 256
rejecting a session 148
releasing a suspended session 483
replying to a confirmation request 389
requesting permission to enter send state 450
resetting the continuation mode 375
restoring a mode 164
sending an error notification 431
sending data 396
sending data and flushing the buffer 418
sending data conditionally 381, 408
sending expedited data on a full-duplex session 457
specific-mode RECEIVEs 348
suspending a subsequent conversation 487
terminating a session 359
terminating a session due to errors 369
terminating a session due to protocol errors or cleanup deactivation 364
unconditional deallocation 119
unconditional deallocation that includes data 109
accepting a session macroinstruction 132
accepting a session, macroinstruction 132
access method control block (ACB) keyword
accepting a session 134
allocation request for contention-winner session 30
allocation request for specific session 20
allocation request, conditional, without wait 51
allocation requests that can be queued 9
any-mode RECEIVEs 315
CNOS requests 140
conditional deallocation 90
conditional deallocation that includes data 98
deallocation due to program errors 63
deallocation due to service errors 69
deallocation due to timer errors 76
deallocation due to user-specified criteria 83
defining LU-mode table values 152
displaying LU-mode table values 158
entering receive state conditionally 220
entering receive state conditionally and including data 228
entering receive state unconditionally 249
entering receive state unconditionally and including data 239
flushing the buffer 443
immediate allocation request 41
indicating the beginning of a synchronization exchange 492
indicating the end of a synchronization exchange 496
accessibility 615
activating a session, macroinstruction for 132
ACTSESS qualify value 132
allocating a conversation
APPCCMD CONTROL=ALLOC macroinstruction
macroinstruction syntax and operands 6, 38
session assignment algorithm
conditional request, without wait 48
immediate requests 38
request for specific session 17
requests for contention-winner sessions 27
requests that can be queued 6
ALLOCCD qualify value
session assignment algorithm 6
syntax and operands 6
any-mode RECEIVEs, macroinstruction 312
APPCCMD macroinstruction
coding comments 3
continuation lines 3
AREA keyword,
as pointer to
DEFINE/DISPLAY control block 158
AREA keyword, (continued)
as pointer to (continued)
session limits control block 140
returned value, used with
  obtain status on information available on a specified
  conversation 516
  obtain status on information from any active
  conversation 501
  obtain status on information immediately available
  from any active conversation 506
  obtain status on information immediately available
  on a specified conversation 510
used with
  accepting a session 134
  allocation request for contention-winner session 30
  allocation request for specific session 20
  allocation request that can be queued 9
  allocation request, conditional, without wait 51
  any-mode RECEIVEs 315
  CNOS requests 140
  conditional deallocation that includes data 98
  deallocation due to program errors 63
  deallocation due to service errors 70
  deallocation due to timer errors 77
  deallocation due to user-specified criteria 84
  defining LU-mode table values 152
  displaying LU-mode table values 158
  entering receive state conditionally and including
data 228
  entering receive state unconditionally and including
data 239
  immediate allocation request 41
  queued deallocation due to program, service, timer,
or user errors 127
  receive normal information immediately available
  from any conversation 326
  receive normal information immediately available
  on a specific conversation 337
  receiving an FMH-5 298
  receiving expedited data from the specified
  conversation 278
  receiving expedited data immediately available in
  any-mode 263
  receiving expedited data immediately from the
  specified conversation 271
  receiving expedited data in any-mode 256
  restoring a mode 164
  specific-mode RECEIVEs 348
ARG keyword
  relationship to CID 148
  used with accepting a session 135
  used with rejecting a session 148
AVFA RPL field
  returned value, used with
    allocation request for contention-winner session 34
    allocation request for specific session 23
    allocation request that can be queued 12
    allocation request, conditional, without wait 55
    CNOS request 143
    immediate allocation request 44
B
becoming receiving LU 218, 247
becoming sender
  requesting permission to enter send state
macroinstruction 448
BIND image
  DSECT 569
  relationship to
  RESTORE 162
BNDAREA DSECT
  profile 1 572
  profile 2 574
  profile 3 574
  profile 4 575
  profile 6 576
BRANCH keyword, used with
  accepting a session 135
  allocation request for content-winner session 30
  allocation request for specific session 20
  allocation request that can be queued 9
  allocation request, conditional, without wait 51
  any-mode RECEIVEs 315
  CNOS requests 140
  conditional deallocation 90
  conditional deallocation that includes data 99
  deallocation due to program errors 63
  deallocation due to service errors 70
  deallocation due to timer errors 77
  deallocation due to user-specified criteria 84
  defining LU-mode table values 153
  displaying LU-mode table values 158
  entering receive state conditionally 220

AREALEN keyword, used with (continued)
  obtain status on information immediately available on a
  specified conversation 511
  receive normal information immediately available from
  any conversation 326
  receive normal information immediately available on a
  specific conversation 337
  receiving an FMH-5 298
  receiving expedited data from the specified
  conversation 278
  receiving expedited data immediately available in
  any-mode 263
  receiving expedited data immediately from the specified
  conversation 271
  receiving expedited data in any-mode 256
  restoring a mode 164
  specific-mode RECEIVEs 348
BRANCH keyword, used with (continued)
entering receive state conditionally and including data 229
entering receive state unconditionally 250
entering receive state unconditionally and including data 240
flushing the buffer 443
immediate allocation request 41
indicating the beginning of a synchronization exchange 492
indicating the end of a synchronization exchange 496
obtain status on information available on a specified conversation 517
obtain status on information from any active conversation 501
obtain status on information immediately available from any active conversation 506
obtain status on information immediately available on a specified conversation 511
queued deallocation due to program, service, timer, or user errors 128
receive normal information immediately available from any conversation 326
receive normal information immediately available on a specific conversation 337
receiving an FMH-5 298
receiving expedited data from the specified conversation 279
receive normal information immediately available on a specific conversation 337
specific-mode RECEIVES 349
CD keyword, used with (continued)
receive normal information immediately available on a specific conversation 337
specific-mode RECEIVES 349
CGID
keyword, used with
allocation request for specific session 20
terminating a session due to protocol errors or cleanup deactivation 365
CGID RPL field
returned value, used with
allocation request for contention-winner session 34
allocation request that can be queued 13
allocation request, conditional, without wait 55
immediate allocation request 44
receiving an FMH-5 300
change number of sessions (CNOS) macroinstruction 137
changing continuation modes
use of RESETRCV macroinstruction variation 372
CHECK control value 59
CHECK macroinstruction 59
CNOS (change number of sessions) macroinstruction 137
CNOS session limits data structure—ISTSLCNS DSECT 586
coding comments 2, 3
Communications Server for z/OS, online information xv
CONFIRM qualify value
DEALLOC CONTROL value 88
PREPRCV CONTROL value 218
SEND CONTROL value 379
confirmation request
replying to a confirmation request macroinstruction, used with 387
sending an error notification macroinstruction, used with 427
CONFRMD qualify value 387
CONFTEXT keyword, used with
accepting a session 135
CONMODE keyword, used with
allocation request for contention-winner session 31
allocation request for specific session 20
allocation request that can be queued 9
allocation request, conditional, without wait 51
any-mode RECEIVES 315
conditional deallocation 90
conditional deallocation that includes data 99, 110, 120
entering receive state conditionally 220
entering receive state conditionally and including data 229
entering receive state unconditionally 250
entering receive state unconditionally and including data 240
flushing the buffer 443
immediate allocation request 41
ISTRPL6 526
receive normal information immediately available from any conversation 326
receive normal information immediately available on a specific conversation 337
receiving expedited data from the specified conversation 279
receiving expedited data immediately available in any-mode 263
receiving expedited data immediately from the specified conversation 271
flushing the buffer 443
immediate allocation request 41
ISTRPL6 526
receive normal information immediately available from any conversation 326
C
CONMODE keyword, used with (continued)
  receiving expedited data in any-mode 256
  replying to a confirmation request 390
  requesting permission to enter send state 450
  resetting the continuation mode 375
  sending an error notification 431
  sending data 397
  sending data and flushing the buffer 419
  sending data conditionally 382, 408
  sending expedited data on a full-duplex session 458
  specific-mode RECEIVES 349

CONSTATE RPL field
  returned value, used with
    allocation request for contention-winner session 34
    allocation request for specific session 23
    allocation request that can be queued 13
    allocation request, conditional, without wait 55
    any-mode RECEIVES 317
    conditional deallocation 92
    conditional deallocation that includes data 102
    deallocation due to program errors 64
    deallocation due to service errors 71
    deallocation due to timer errors 78
    deallocation due to user-specified criteria 85
    entering receive state conditionally 222
    entering receive state conditionally and including data 232
    entering receive state unconditionally 251
    entering receive state unconditionally and including data 243
    flushing the buffer 445
    immediate allocation request 44
    obtain status on information available on a specified conversation 518
    obtain status on information immediately available on a specified conversation 512
    queued deallocation due to program, service, timer, or user errors 129
    receive normal information immediately available from any conversation 328
    receive normal information immediately available on a specific conversation 340
    receiving an FMH-5 300
    receiving expedited data from the specified conversation 280
    receiving expedited data immediately available in any-mode 265
    receiving expedited data immediately from the specified conversation 273
    replying to confirmation request 391
    requesting permission to enter send state 452
    resetting the continuation mode 377
    sending an error notification 434
    sending data 400
    sending data and flushing the buffer 422
    sending data conditionally 383, 411
    sending expedited data on full-duplex session 460
    specific-mode RECEIVES 351
    terminating a session 360
    unconditional deallocation 121
    unconditional deallocation that includes data 113
    contention winner
    requests for allocation 27

CONTROL keyword
  ALLOC value
    ALLOCD QUALIFY value 6
    CONVGRP QUALIFY value 17
    CONWIN QUALIFY value 27
    IMMED QUALIFY value 38
    WHENFREE QUALIFY value 48, 207
  CHECK value 59
  DEALLOC value
    ABNDPROG QUALIFY value 60
    ABNDUSER QUALIFY value 81
    ABNDUSER QUALIFY value 81
    CONFIRM quality value 88
    DATACON QUALIFY value 96
    DATAFLU QUALIFY value 107
    FLUSH QUALIFY value 117
  DEALOCQ value
    ABNDPROG QUALIFY value 124
    ABNDUSER QUALIFY value 124
    ABNDUSER QUALIFY value 124
  OPRCTL value
    ACTSESS QUALIFY value 132
    CNOS QUALIFY value 137
    DACTSESS QUALIFY value 146
    DEFINE QUALIFY value 150
    DISPLAY QUALIFY value 156
    RESTORE QUALIFY value 162
  PREPRCV value
    CONFIRM QUALIFY value 218
    DATACON QUALIFY value 226
    DATAFLU QUALIFY value 237
    FLUSH QUALIFY value 247
  RCVEXPD value
    ANY QUALIFY value 253
    IANY QUALIFY value 261
    ISPEC QUALIFY value 268
    SPEC QUALIFY value 276
  RCVFMH5 value 295
  RECEIVE value
    ANY QUALIFY value 312
    IANY QUALIFY value 323
    ISPEC QUALIFY value 334
    SPEC QUALIFY value 345
  REJECT value
    CONV QUALIFY value 357
    CONVGRP QUALIFY value 362
    SESSION QUALIFY value 367
  RESERV value
    CONFIRM quality value 372
  SEND value
    CONFIRM QUALIFY value 379
    CONFRMD QUALIFY value 387
    DATA QUALIFY value 394
    DATACON QUALIFY value 405
    DATAFLU QUALIFY value 416
    ERROR QUALIFY value 427
    FLUSH QUALIFY value 441
    RQSEND QUALIFY value 448
  SENSEXPD value 454
  SETSESSION value
    RESUME QUALIFY value 481
    SUSPEND QUALIFY value 485
conversation allocation
APPCCMD CONTROL=ALLOC macroinstruction
macroinstruction syntax and operands 6, 38
session assignment algorithm
conditional request, without wait 48, 207
immediate requests 38
request for specific session 17
requests for contention-winner sessions 27
requests that can be queued 6

conversation data, receiving
APPCCMD macroinstructions used 312, 345

conversation data, sending
APPCCMD macroinstructions used 379, 448
buffer list (OPTCD=BUFFLST) considerations, used with
conditional deallocation that includes data 100
entering receive state conditionally and including data 231
sending data 101, 399
sending data and flushing the buffer 231, 242, 421
sending data conditionally 410
sending expedited data on a full-duplex session 460
unconditional deallocation that includes data 112
flushing the buffer, used with
conditional deallocation 88
conditional deallocation that includes data 96
entering receive state conditionally and including data 226
entering receive state unconditionally and including data 226
entering receive state unconditionally 247
entering receive state unconditionally and including data 237
flushing the buffer 441
sending data and flushing the buffer 416
sending data conditionally 379, 405
unconditional deallocation 117
unconditional deallocation that includes data 107
macroinstructions for sending new data, used with
conditional deallocation that includes data 96
entering receive state conditionally and including data 226
entering receive state unconditionally and including data 237
sending data 394
sending data and flushing the buffer 416
sending data conditionally 405
unconditional deallocation that includes data 107
sending expedited data on a full-duplex session 454

conversation deallocation
APPCCMD CONTROL=DEALLOC
macroinstruction syntax and operands 60, 117
APPCCMD CONTROL=DEALLOQC
macroinstruction syntax and operands 124
use of macroinstruction 124

conversation identifier
keyword, used with
conditional deallocation 91
conditional deallocation that includes data 99
deallocation due to program errors 63
deallocation due to service errors 70
deallocation due to timer errors 77
deallocation due to user-specified criteria 84
entering receive state conditionally 221
entering receive state conditionally and including data 229
entering receive state unconditionally 250
entering receive state unconditionally and including data 240
flushing the buffer 444
indicating the beginning of a synchronization exchange 492
obtain status on information available on a specified conversation 517
obtain status on information immediately available on a specified conversation 511
queued deallocation due to program, service, timer, or user errors 128
receive normal information immediately available on a specific conversation 338
replying to a confirmation request 390
requesting permission to enter send state 451
resetting the continuation mode 375
sending an error notification 431
sending data 398
sending data and flushing the buffer 420
sending data conditionally 382, 409
sending expedited data on a full-duplex session 458
specific-mode RECEIVES 349
suspending a subsequent conversation 488
terminating a session 359
unconditional deallocation 120
unconditional deallocation that includes data 110
returned value, used with
allocation request for contention-winner session 35
allocation request for specific session 23
allocation request that can be queued 13
allocation request, conditional, without wait 55
any-mode RECEIVES 318
immediate allocation request 45
receive normal information immediately available from any conversation 329
receiving an FMH-5 300
receiving expedited data in any-mode 259

conversation security
conversation level 576
CONVGRP qualify value
session assignment algorithm 17
CONVID
keyword, used with
conditional deallocation 91
conditional deallocation that includes data 99
deallocation due to program errors 63
deallocation due to service errors 70
deallocation due to timer errors 77
deallocation due to user-specified criteria 84
entering receive state conditionally 221
CONVID (continued)
keyword, used with (continued)
entering receive state conditionally and including data 229
entering receive state unconditionally 250
flushing the buffer 444
indicating the beginning of a synchronization exchange 492
obtain status on information immediately available on a specified conversation 517
obtain status on information immediately available on a specified conversation 511
queued deallocation due to program, service, timer, or user errors 128
receive normal information immediately available on a specific conversation 338
replying to a confirmation request 390
requesting permission to enter send state 451
resetting the continuation mode 375
sending an error notification 431
sending data 398
sending data and flushing the buffer 420
sending data conditionally 409
sending expedited data on a full-duplex session 458
specific-mode RECEIVEs 349
suspending a subsequent conversation 488
terminating a session 359
unconditional deallocation 120
unconditional deallocation that includes data 110
used with entering receive state unconditionally and including data 240
returned value, used with
allocation request for specific session 23
allocation request that can be queued 13
allocation request, conditional, without wait 55
any-mode RECEIVEs 318
immediate allocation request 45
receive normal information immediately available from any conversation 329
receiving an FMH-5 300
receiving expedited data from the specified conversation 279
receiving expedited data immediately available in any-mode 266
receiving expedited data immediately available from the specified conversation 271
receiving expedited data immediately from the specified conversation 271
receiving expedited data in any-mode 257
relying to a confirmation request 390
resetting the continuation mode 375
sending an error notification 432
sending data 398
sending data and flushing the buffer 420
sending data conditionally 409
sending expedited data on full-duplex session 458
specific-mode RECEIVEs 349
cross-memory mode, restrictions 59
CRYPT
keyword, used with
conditional deallocation that includes data 100
entering receive state 230
entering receive state unconditionally 241
sending data 398
sending data and flushing the buffer 420
sending data conditionally 409
unconditional deallocation that includes data 111
CRYPTLVL RPL field
returned value, used with
allocation request for contention-winner session 35
allocation request for specific session 24
allocation request that can be queued 13
allocation request, conditional, without wait 55
immediate allocation request 45
receiving an FMH-5 301
D
DACTSESS qualify value
used with terminating a session 146
DATA qualify value 394, 454
data truncation
sending an error notification macroinstruction 427
data, receiving
APPCCMD macroinstructions used 312, 345
data, sending
APPCCMD macroinstructions used
data, sending (continued)
  APPCCMD macroinstructions used (continued)
    syntax and operands 379, 448
  buffer list (OPTCD=BUFFLST) considerations
    conditional deallocation that includes data 100
    entering receive state conditionally and including
    data 231
    sending data 101, 399
    sending data and flushing the buffer 231, 242, 421
    sending data conditionally 410
    sending expedited data on a full-duplex session 460
    unconditional deallocation that includes data 112
    flushing the buffer, used with
    conditional deallocation 88
    conditional deallocation that includes data 96
    entering receive state conditionally 218
    entering receive state conditionally and including
    data 226
    entering receive state unconditionally 247
    entering receive state unconditionally and including
    data 237
    flushing the buffer 441
    sending data and flushing the buffer 416
    sending data conditionally 379, 405
    unconditional deallocation 117
    unconditional deallocation that includes data 107
  macroinstructions for sending new data, used with
  conditional deallocation that includes data 96
  entering receive state conditionally and including
  data 226
  entering receive state unconditionally and including
  data 237
  sending data 394
  sending data and flushing the buffer 416
  sending data conditionally 405
  unconditional deallocation that includes data 107
  sending expedited data on a full-duplex session 454

DATACON qualify value
  DEALLOC CONTROL value 96
  PREPRCV CONTROL value 226
  SEND CONTROL value 405

DATAFLU qualify value
  DEALLOC CONTROL value 107
  PREPRCV CONTROL value 237
  SEND CONTROL value 416

deactivating sessions
  used with terminating a session macroinstruction 146

deactivation type code (DEACTYP)
  keyword, used with
  terminating session due to error 370
  terminating session due to protocol error or cleanup
  deactivation 365

DEACTYP (deactivation type code)
  keyword, used with
  terminating session due to error 370
  terminating session due to protocol error or cleanup
  deactivation 365

deallocating a conversation
  APPCCMD CONTROL=DEALLOC
    macroinstruction syntax and operands 60, 81
  APPCCMD CONTROL=DEALLOQC
    macroinstruction syntax and operands 124
    use of macroinstruction 124

deallocating conversations
  macroinstruction forms of CONTROL=DEALLOC 60, 117
  deallocation, abnormal
    LU Services errors (ABNDSERV) 67, 124
    reference material (macroinstruction syntax and
    operands) 60, 81
    timing errors (ABNDTIME) 74, 124
    transaction program errors (ABNDPROG) 60, 124
    user-defined errors (ABNDUSER) 81, 124

DEFINE qualify value
  syntax and operands 150
  defining LU-mode table values 150
  design considerations for LU 6.2
    restoring a mode 162
  disability 615

DISPLAY qualify value
  syntax and operands 156
  displaying LU-mode data 156

DNS, online information xvi

DSECT
  ISTDBIND 569
  ISTFMS 577
  ISTRPL6X 579
  ISTSLCNS 586
  ISTSLD 587
  ISTSREST 588
  ISTSTATD 589
  ISTUSFBC 589

E

ECB (event control block)
  keyword, used with
  accepting a session 135
  allocation request for contention-winner session 31
  allocation request for specific session 21
  allocation request that can be queued 10
  allocation request, conditional, without wait 52
  any-mode RECEIVES 316
  CNOS requests 140
  conditional deallocation 91
  conditional deallocation that includes data 100
  deallocation due to program errors 63
  deallocation due to service errors 70
  deallocation due to timer errors 77
  deallocation due to user-specified criteria 84
  defining LU-mode table values 153
  displaying LU-mode table values 158
  entering receive state conditionally 221
  entering receive state conditionally and including
  data 230
  entering receive state unconditionally 251
  entering receive state unconditionally and including
  data 241
   flushing the buffer 444
  immediate allocation request 42
  indicating the beginning of a synchronization
  exchange 492
  indicating the end of a synchronization exchange
  497
  obtaining status on information available on a
  specified conversation 517
  obtaining status on information from any active
  conversation 501
ECB (event control block) (continued)
keyword, used with (continued)
  obtain status on information immediately available from any active conversation 506
  obtain status on information immediately available on a specified conversation 511
  queued deallocation due to program, service, timer, or user errors 128
  receive normal information immediately available from any active conversation 327
  receive normal information immediately available on a specific conversation 338
  receiving an FMH-5 299
  receiving expedited data from the specified conversation 279
  receiving expedited data immediately available in any-mode 264
  receiving expedited data immediately available from the specified conversation 272
  receiving expedited data in any-mode 257
  rejecting a session 148
  releasing a suspended session 483
  replying to a confirmation request 391
  requesting permission to enter send state 451
  resetting the continuation mode 376
  restoring a mode 165
  sending an error notification 432
  sending data 398
  sending data and flushing the buffer 420
  sending data conditionally 382
  sending expedited data on a full-duplex session 459
  specific-mode RECEIVES 350
  suspending a subsequent conversation 488
  terminating a session 359
  terminating a session due to errors 370
  terminating a session due to protocol errors or cleanup deactivation 365
  unconditional dealloaction 121
  unconditional deallocation that includes data 111
  entering receive state conditionally 221
  entering receive state conditionally and including data 230
  entering receive state unconditionally 251
  entering receive state unconditionally and including data 241
  flushing the buffer 444
  immediate allocation request 42
  indicating the beginning of a synchronization exchange 492
  indicating the end of a synchronization exchange 497
  obtain status on information available on a specified conversation 517
  obtain status on information from any active conversation 501
  obtain status on information immediately available from any active conversation 506
  obtain status on information immediately available on a specific conversation 511
  queued deallocation due to program, service, timer, or user errors 128
  receive normal information immediately available from any conversation 327
  receive normal information immediately available on a specific conversation 338
  receiving an FMH-5 299
  receiving expedited data from the specified conversation 279
  receiving expedited data immediately available in any-mode 264
  receiving expedited data immediately available from the specified conversation 272
  receiving expedited data in any-mode 257
  rejecting a session 148
  releasing a suspended session 483
  replying to a confirmation request 391
  requesting permission to enter send state 451
  resetting the continuation mode 376
  restoring a mode 165
  sending an error notification 432
  sending data 398
  sending data and flushing the buffer 420
  sending data conditionally 382
  sending expedited data on a full-duplex session 459
  specific-mode RECEIVES 350
  suspending a subsequent conversation 488
  terminating a session 359
  terminating a session due to errors 370
  terminating a session due to protocol errors or cleanup deactivation 365
  unconditional dealloaction 121
  unconditional deallocation that includes data 111
  entering RECEIVE state
  PREPRCV macroinstructions (syntax and operands) 218, 247
  entering SEND state
  requesting permission to enter send state
  macroinstruction 448
  ERROR qualify value 427
  event control block (ECB) (continued)
keyword, used with (continued)
  accepting a session 135
  allocation request for contention-winner session 31
  allocation request for specific session 21
  allocation request that can be queued 10
  allocation request, conditional, without wait 52
  any-mode RECEIVES 316
  CNOS requests 140
  conditional deallocation 91
  conditional deallocation that includes data 100
  deallocation due to program errors 63
  deallocation due to service errors 70
  deallocation due to timer errors 77
  deallocation due to user-specified criteria 84
  defining LU-mode table values 153
  displaying LU-mode table values 158
  EXIT keyword, used with
  accepting a session 135
  allocation request for contention-winner session 32
  allocation request for specific session 21
  allocation request that can be queued 10
  allocation request, conditional, without wait 52
  any-mode RECEIVES 316
  CNOS requests 140
EXIT keyword, used with (continued)
conditional deallocation 92
conditional deallocation that includes data 100
deallocation due to program errors 63
deallocation due to service errors 70
deallocation due to timer errors 77
deallocation due to user-specified criteria 84
defining LU-mode table values 153
displaying LU-mode table values 159
entering receive state conditionally 221
entering receive state conditionally and including data 230
entering receive state unconditionally 251
entering receive state unconditionally and including data 241
flushing the buffer 444
immediate allocation request 42
indicating the beginning of a synchronization exchange 493
indicating the end of a synchronization exchange 497
queued deallocation due to program, service, timer, or user errors 257
receive normal information immediately available from any conversation 327
receive normal information immediately available on a specific conversation 338
receiving an FMH-5 299
receiving expedited data from the specified conversation 280
receiving expedited data immediately available in any-mode 264
rejecting a session 148
releasing a suspended session 483
replying to a confirmation request 391
requesting permission to enter send state 451
resetting the continuation mode 376
restoring a mode 165
sending an error notification 432
sending data 398
sending data and flushing the buffer 421
sending data conditionally 384, 410
specific-mode RECEIVES 350
suspending a subsequent conversation 488
terminating a session 360
terminating a session due to errors 370
terminating a session due to protocol errors or cleanup deactivation 365
unconditional deallocation 121
unconditional deallocation that includes data 111

EXPDLEN RPL field
returned value, used with (continued)
receive normal information immediately available on a specific conversation 340
receiving expedited data from the specified conversation 281
receiving expedited data immediately available in any-mode 266
receiving expedited data immediately from the specified conversation 273
receiving expedited data in any-mode 259
sending an error notification 434
sending data 401
sending data and flushing the buffer 423
sending data conditionally 384, 412
sending expedited data on full-duplex session 461
specific-mode RECEIVES 352
unconditional deallocation 122
unconditional deallocation that includes data 113

EXPDRCV RPL field
returned value, used with (continued)
receive normal information immediately available from any conversation 329
receive normal information immediately available on a specific conversation 340
receiving expedited data from the specified conversation 281
receiving expedited data immediately available in any-mode 266
receiving expedited data immediately from the specified conversation 273
receiving expedited data in any-mode 259
sending an error notification 434
sending data 401
sending data and flushing the buffer 423
sending data conditionally 384, 412
sending expedited data on full-duplex session 461
specific-mode RECEIVES 352
unconditional deallocation 122
unconditional deallocation that includes data 113

F
FDB2 RPL field
returned value, used with
accepting a session 136
allocation request for contention-winner session 35
allocation request for specific session 24
allocation request that can be queued 13
allocation request, conditional, without wait 56
any-mode RECEIVES 318
CNOS requests 143
conditional deallocation 93
FDB2 RPL field (continued)
conditional deallocation that includes data 102
deallocation due to program errors 65
deallocation due to service errors 72
deallocation due to timer errors 79
deallocation due to user-specified criteria 86
defining LU-mode table values 154
displaying LU-mode table values 160
entering receive state conditionally 223
entering receive state conditionally and including data 233
entering receive state unconditionally 252
entering receive state unconditionally and including data 243
flushing the buffer 446
immediate allocation request 45
indicating the beginning of a synchronization exchange 493
indicating the end of a synchronization exchange 498
obtain status on information available on a specified conversation 519
obtain status on information from any active conversation 502
obtain status on information immediately available on a specified conversation 513
queued deallocation due to program, service, timer, or user errors 130
receive normal information immediately available from any conversation 329
receive normal information immediately available on a specific conversation 340
receiving an FMH-5 301
receiving expedited data from the specified conversation 281
receiving expedited data immediately available in any-mode 266
receiving expedited data immediately from the specified conversation 273
receiving expedited data in any-mode 259
rejecting a session 149
releasing a suspended session 484
replying to a confirmation request 392
requesting permission to enter send state 453
resetting the continuation mode 378
restoring a mode 166
sending an error notification 435
sending data 401
sending data and flushing the buffer 423
sending data conditionally 384, 412
sending expedited data on full-duplex session 461
specific-mode RECEIVES 352
suspending a subsequent conversation 489
terminating a session 361
terminating a session due to errors 371
terminating a session due to protocol errors or cleanup deactivation 366
unconditional deallocation 122
unconditional deallocation that includes data 113
feedback code data structure—ISTUSFBC DSECT 589
feedback information request-to-send indicator 448
FILL keyword...

FILL keyword (continued)
receive normal information immediately available on a specific conversation 338
used with ISTRPL6 527
used with specific-mode RECEIVES 350
FLUSH qualify value
DEALLOC CONTROL value 117
PREPRCV CONTROL value 247
SEND CONTROL value 441
flushing the send buffer
flushing the buffer macroinstruction 441
used with entering receive state unconditionally macroinstruction 247
used with unconditional deallocation macroinstruction 117
FMH-5 DSECT 577
FMHSLEN RPL field returned value, used with
allocation request for contention-winner session 35
allocation request for specific session 24
allocation request that can be queued 14
allocation request, conditional, without wait 56
any-mode RECEIVES 318
conditional deallocation 93
conditional deallocation that includes data 102
deallocation due to program errors 65
deallocation due to service errors 72
deallocation due to timer errors 79
deallocation due to user-specified criteria 86
entering receive state conditionally 223
entering receive state conditionally and including data 233
entering receive state unconditionally 252
entering receive state unconditionally and including data 243
flushing the buffer 446
immediate allocation request 45
indicating the beginning of a synchronization exchange 493
indicating the end of a synchronization exchange 498
obtain status on information available on a specified conversation 519
obtain status on information from any active conversation 502
obtain status on information immediately available on a specified conversation 513
queued deallocation due to program, service, timer, or user errors 130
receive normal information immediately available from any conversation 329
receive normal information immediately available on a specific conversation 340
receiving an FMH-5 301
receiving expedited data from the specified conversation 281
receiving expedited data immediately available in any-mode 266
receiving expedited data immediately from the specified conversation 274
receiving expedited data in any-mode 259
replying to a confirmation request 392
requesting permission to enter send state 453
resetting the continuation mode 378
sending an error notification 435
FMH5LEN RPL field (continued)
returned value, used with (continued)
sending data 401
sending data and flushing the buffer 423
sending data conditionally 384, 412
sending expedited data on full-duplex session 461
specific-mode RECEIVEs 352
terminating a session 361
terminating a session due to errors 371
terminating a session due to protocol errors or cleanup deactivation 366
unconditional deallocation 122
unconditional deallocation that includes data 114

FMH5RCV RPL field
returned value, used with
allocation request for contention-winner session 35
allocation request for specific session 23
allocation request that can be queued 13
any-mode RECEIVEs 319
conditional deallocation 93
conditional deallocation that includes data 102
deallocation due to program errors 65
deallocation due to service errors 72
deallocation due to timer errors 79
deriving receive state conditionally 223
deriving receive state conditionally and including data 233
deriving receive state unconditionally 252
deriving receive state unconditionally and including data 243
flushing the buffer 446
immediate allocation request 45
obtain status on information available on a specified conversation 519
obtain status on information from any active conversation 502
obtain status on information immediately available from any active conversation 507
obtain status on information immediately available on a specified conversation 513
queued deallocation due to program, service, timer, or user errors 130
receive normal information immediately available from any conversation 329
receive normal information immediately available on a specific conversation 341
receiving an FMH-5 301
receiving expedited data from the specified conversation 281
receiving expedited data immediately available in any-mode 266
receiving expedited data immediately from the specified conversation 274
receiving expedited data in any-mode 259
replying to a confirmation request 392
requesting permission to enter send state 453
resetting the continuation mode 378
sending an error notification 435
sending data 401
sending data and flushing the buffer 423
sending data conditionally 384, 412
sending expedited data on full-duplex session 461

FMH5RCV RPL field (continued)
returned value, used with (continued)
specific-mode RECEIVEs 352
terminating a session 361
terminating a session due to errors 371
terminating a session due to protocol errors or cleanup deactivation 366
unconditional deallocation 122
unconditional deallocation that includes data 114

G
global variables
declaring and setting with ISTGAPPC macroinstruction 520
table of 521–523

I
IBM Software Support Center, contacting x identifier, conversation
keyword, used with
conditional deallocation 91
conditional deallocation that includes data 99
deallocation due to program errors 63
deallocation due to service errors 70
deallocation due to timer errors 77
deallocation due to user-specified criteria 84
deriving receive state conditionally 221
deriving receive state conditionally and including data 229
deriving receive state unconditionally 250
deriving receive state unconditionally and including data 240
flushing the buffer 444
indicating the beginning of a synchronization exchange 492
obtain status on information available on a specified conversation 517
obtain status on information immediately available on a specified conversation 511
queued deallocation due to program, service, timer, or user errors 128
receive normal information immediately available on a specific conversation 338
replying to a confirmation request 390
requesting permission to enter send state 451
resetting the continuation mode 375
sending an error notification 431
sending data 398
sending data and flushing the buffer 420
sending data conditionally 382, 409
sending expedited data on a full-duplex session 458
specific-mode RECEIVEs 349
suspending a subsequent conversation 488
terminating a session 359
unconditional deallocation 120
unconditional deallocation that includes data 110
returned value, used with
allocation request for contention-winner session 35
allocation request for specific session 23
allocation request that can be queued 13
identifier, conversation (continued)
  returned value, used with (continued)
    allocation request, conditional, without wait 55
    any-mode RECEIVES 318
    immediate allocation request 45
    receive normal information immediately available
      from any conversation 329
    receiving an FMH-5 300
    receiving expedited data in any-mode 259
IMMED qualify value
  session assignment algorithm 38
  use of 38
Information APARs xiii
Internet, finding z/OS information online xv
ISTDBIND DSECT 569
ISTFM5 DSECT 577
ISTGAPPC macroinstruction 520
ISTRPL6 macroinstruction 523
ISTRPL6X DSECT 579
ISTSLCNS DSECT 586
ISTSLD DSECT 587
ISTSREST DSECT 588
ISTUSFBC DSECT 589

K
keyboard 615

L
license, patent, and copyright information 619
limited resource
  session 577
LIST
  keyword, used with
    ISTRPL6 528
    restoring a mode 165
LOCKS
  keyword, used with
    entering receive state conditionally 221
    entering receive state conditionally and including data 230
    ISTRPL6 528
LOGMODE
  keyword, used with
    allocation request for contention-winner session 32
    allocation request that can be queued 10
    allocation request, conditional, without wait 52
    CNOS requests 141
    defining LU-mode table values 153
    displaying LU-mode table values 159
    immediate allocation request 42
    ISTRPL6 528
    restoring a mode 165
    returned value, used with
      allocation request for specific session 24
      receiving an FMH-5 301
LOGRCV RPL field (continued)
  returned value, used with (continued)
    entering receive state conditionally and including data 233
    entering receive state unconditionally and including data 244
    receive normal information immediately available
      from any conversation 330
    receiving expedited data in any-mode 259
    sending an error notification 435
    sending data 401
    sending data and flushing the buffer 423
    sending data conditionally 384, 412
    specific-mode RECEIVES 352

LU 6
  restoring a mode 162
LU-mode table
  changing information in table 150
  querying information in table 156
LUAFFIN
  keyword, used with
    allocation request for a contention-winner session 32
    allocation request for a specific session 21
    allocation request that can be queued 10
    CNOS requests 141
    defining LU-mode table values 153
    displaying LU-mode table values 159
    immediate allocation request 43
    ISTRPL6 528
    restoring a mode 165
    returned value, used with
      allocation request for specific session 25
      receiving an FMH-5 301

M
macroinstruction
  operands 2
  syntax 1
mainframe
  education xiii
mode
  cross-memory, restrictions 59
  restoring 162

N
NAMEUSE
  keyword, used with
    allocation request for contention-winner session 32
NAMEUSE (continued)
keyword, used with (continued)
 allocation request for specific session 21
 allocation request that can be queued 11
 allocation request, conditional, without wait 53
 CNOS requests 141

NETID
keyword, used with
 allocation request for contention-winner session 33
 allocation request that can be queued 11
 allocation request, conditional, without wait 53
 CNOS requests 141
 defining LU-mode tables 153
 displaying LU-mode table values 159
 immediate allocation request 43
 ISTRPL6 528
 restoring a mode 166
 returned value, used with
 allocation request for specific session 25
 receiving an FMH-5 301

O
operand format 2
operand notation 2

OPTCD keyword, used with
 accepting a session 135
 allocation request for contention-winner session 33
 allocation request for specific session 22
 allocation request that can be queued 11
 allocation request, conditional, without wait 53
 any-mode RECEIVES 316
 CNOS requests 142
 conditional deallocation 92
 conditional deallocation that includes data 100
 deallocation due to program errors 63
 deallocation due to service errors 70
 deallocation due to timer errors 77
 defining LU-mode table values 154
 displaying LU-mode table values 159
 entering receive state conditionally 222
 entering receive state conditionally and including data 231
 entering receive state unconditionally 251
 entering receive state unconditionally and including data 241
 flushing the buffer 444
 immediate allocation request 43
 indicating the beginning of a synchronization exchange 493
 indicating the end of a synchronization exchange 497
 obtain status on information from any active conversation 501
 obtain status on information immediately available from any active conversation
 obtain status on information available on a specified conversation 517
 obtain status on information immediately available on a specified conversation 511
 queued deallocation due to program, service, timer, or user errors 128
 receiving normal information immediately available from any conversation 327

OPTCD keyword, used with (continued)
 receive normal information immediately available on a specific conversation 339
 receiving an FMH-5 299
 receiving expedited data from the specified conversation 280
 receiving expedited data immediately available in any-mode 264
 receiving expedited data immediately from the specified conversation 272
 receiving expedited data in any-mode 257
 rejecting a session 148
 replying to a confirmation request 391
 requesting permission to enter send state 451
 resetting the continuation mode 376
 restoring a mode 166
 sending an error notification 432
 sending data 398
 sending data and flushing the buffer 421
 sending data conditionally 383
 sending expedited data on a full-duplex session 459
 specific-mode RECEIVES 350
 suspending a subsequent conversation 488
 terminating a session 360
 terminating a session due to errors 370
 terminating a session due to protocol errors or cleanup deactivation 365
 unconditional deallocation 121
 unconditional deallocation that includes data 111

P
parameters, session
 DSECT 569
 relationship to
 RESTORE 162

PIP (program initialization parameters) data
 DSECT 577
 preparing to receive data 218, 247
 PREPRCV macroinstructions (syntax and operands) . 218, 247
 prerequisite information xiii

program initialization parameters (PIP) data
 DSECT 577
 PRSISTVP RPL field
 returned value, used with
 allocation request for contention-winner session 36
 allocation request for specific session 25
 allocation request that can be queued 14
 allocation request, conditional, without wait 56
 CNOS request 143
 immediate allocation request 46

PSERVIC
 profile 1 572
 profile 2 574
 profile 3 574
 profile 4 575
 profile 6 576

Q
QUALIFY keyword
QUALIFY keyword (continued)
ABNDPROG value 60, 124
ABNDSERV value 67, 124
ABNDTIME value 74, 124
ABNDUSER value 81, 124
ACTSESS value 132
ALLOCD value 6
ANY value
   RCVEXPD CONTROL value 253, 261, 268, 276
   RECEIVE CONTROL value 312, 323
CNOS value 137
CONFIRM value
   CONFIRMED QUALIFY value 387
   DEALLOC CONTROL value 88
   PREPRCV CONTROL value 218
   SEND CONTROL value 379
CONV value 357
CONVGRP value 17, 362
CONWIN value 27
DACTSESS value 146
DATA value
   SEND CONTROL value 394
   SENDEXPD CONTROL value 454
DATACON value
   DEALLOC CONTROL value 96
   PREPRCV CONTROL value 226
   SEND CONTROL value 405
DATAFLU value
   DEALLOC CONTROL value 107
   PREPRCV CONTROL value 237
   SEND CONTROL value 416
DEFINE value 150
DISPLAY value 156
ERROR value 427
FLUSH value
   DEALLOC CONTROL value 117
   PREPRCV CONTROL value 247
   SEND CONTROL value 441
IMMED value 38, 198
ISPEC value 334
RESTORE value 162
RESUME value 481
RQSEND value 448
SESSION value 367
SPEC value 345
SUSPEND value 485
SYNCBEG value 490
SYNCEND value 494
   used with ISTRPL6 529
WHENFREE value 48

R

RCPRI (continued)
   returned value, used with (continued)
   deallocation due to program errors 65
   deallocation due to service errors 72
   deallocation due to timer errors 79
   deallocation due to user-specified criteria 86
   defining LU-mode table values 155
   displaying LU-mode table values 160
   entering receive state conditionally 224
   entering receive state conditionally and including data 234
   entering receive state unconditionally 252
   entering receive state unconditionally and including data 244
   flushing the buffer 446
   immediate allocation request 46
   indicating the beginning of a synchronization exchange 496
   indicating the end of a synchronization exchange 498
   obtain status on information available on a specified conversation 519
   obtain status on information from any active conversation 502
   obtain status on information immediately available from any active conversation 507
   obtain status on information immediately available on a specified conversation 513
   queued deallocation due to program, service, timer, or user errors 130
   receive normal information immediately available from any conversation 330
   receive normal information immediately available on a specific conversation 342
   receiving an FMH-5 301
   receiving expedited data from the specified conversation 282
   receiving expedited data immediately available in any-mode 266
   receiving expedited data immediately from the specified conversation 274
   receiving expedited data in any-mode 259
   rejecting a session 149
   releasing a suspended session 484
   replying to confirmation request 392
   requesting permission to enter send state 453
   resetting the continuation mode 378
   restoring a mode 166
   sending an error notification 436
   sending data 402
   sending data and flushing the buffer 424
   sending data conditionally 385, 413
   sending expedited data on full-duplex session 461
   specific-mode RECEIVES 353
   suspending a subsequent conversation 489
   terminating a session 361
   terminating a session due to errors 371
   terminating a session due to protocol errors or cleanup deactivation 366
   unconditional deallocation 123
   unconditional deallocation that includes data 114

RCSEC
   returned value, used with
   accepting a session 136

643
RCSEC (continued)

  returned value, used with (continued)
  allocation request for specific session 25
  allocation request that can be queued 14
  allocation request, conditional, without wait 56
  allocation requests for contention-winner session 36
  any-mode RECEIVEs 320
  CNOS requests 143
  conditional deallocation 94
  conditional deallocation that includes data 103
  deallocation due to program errors 65
  deallocation due to service errors 72
  deallocation due to timer errors 79
  deallocation due to user-specified criteria 86
  defining LU-mode table values 155
  displaying LU-mode table values 160
  entering receive state conditionally 224
  entering receive state conditionally and including data 234
  entering receive state unconditionally 252
  exiting receive state unconditionally and including data 244
  flushing the buffer 446
  immediate allocation request 46
  indicating the beginning of a synchronization exchange 493
  indicating the end of a synchronization exchange 498
  obtaining status on information available on a specified conversation 519
  obtaining status on information from any active conversation 502
  obtaining status on information immediately available from any active conversation 507
  obtaining status on information immediately available on a specified conversation 513
  queued deallocation due to program, service, timer, or user errors 130
  receive normal information immediately available from any conversation 330
  receive normal information immediately available on a specific conversation 342
  receiving an FMH-5 301
  receiving expedited data from the specified conversation 282
  receiving expedited data immediately available in any-mode 266
  receiving expedited data immediately available in the specified conversation 274
  receiving expedited data in any-mode 259
  rejecting a session 149
  replying to a confirmation request 392
  requesting permission to enter send state 453
  resetting the continuation mode 378
  restoring a mode 167
  sending an error notification 436
  sending data 402
  sending data and flushing the buffer 424
  sending data conditionally 385, 413
  sending expedited data on a full-duplex session 460
  specific-mode RECEIVEs 353
  suspending a subsequent conversation 489

RCSEC (continued)

  returned value, used with (continued)
  allocating a session 361
  terminating a session due to errors 371
  terminating a session due to program errors or cleanup deactivation 366
  unconditional deallocation 123
  unconditional deallocation that includes data 114
  RCVFMH5 control value 295
  RECEIVE state, entering
  PREPRCV macroinstructions (syntax and operands) 218, 247
  receive-any mode
  used with any-mode RECEIVEs macroinstruction 312
  receive-specific mode
  any-mode RECEIVEs macroinstruction, used with 345
  receiving data
  APPCCMD macroinstructions used 312, 345
  RECLEN
  keyword, used with
  accepting a session 136
  allocation request for contention-winner session 33
  allocation request for specific session 22
  allocation request that can be queued 12
  allocation request, conditional, without wait 54
  CNOS requests 142
  conditional deallocation that includes data 101
  deallocation due to program errors 64
  deallocation due to service errors 71
  deallocation due to timer errors 78
  deallocation due to user-specified criteria 85
  defining LU-mode table values 154
  entering receive state conditionally and including data 232
  entering receive state unconditionally and including data 242
  immediate allocation request 43
  queued deallocation due to program, service, timer, or user errors 129
  sending an error notification 433
  sending data 400
  sending data and flushing the buffer 422
  sending data conditionally 411
  sending expedited data on a full-duplex session 460
  unconditional deallocation that includes data 112
  returned value, used with
  any-mode RECEIVEs 320
  displaying LU-mode table values 160
  obtaining status on information available on a specified conversation 519
  obtaining status on information from any active conversation 502
  obtaining status on information immediately available from any active conversation 507
  obtaining status on information immediately available on a specific conversation 513
  receive normal information immediately available from any conversation 330
  receive normal information immediately available on a specific conversation 342
  receiving an FMH-5 302
  receiving expedited data from the specified conversation 282
RECLN (continued)
returned value, used with (continued)
receiving expedited data immediately available in any-mode 267
receiving expedited data immediately from the specified conversation 274
receiving expedited data in any-mode 259
restoring a mode 167
specific-mode RECEIVES 353
reducing session limits 137
register usage
summary 611
request parameter list (RPL) keyword, used with
accepting a session 136
allocation request for contention-winner session 33
allocation request for specific session 22
allocation request that can be queued 12
allocation request, conditional, without wait 54
any-mode RECEIVES 317
CNOS requests 142
conditional deallocation 92
conditional deallocation that includes data 102
CONTROL=CHECK 60
deallocation due to program errors 64
deallocation due to service errors 71
deallocation due to timer errors 78
deallocation due to user-specified criteria 85
defining LU-mode table values 154
displaying LU-mode table values 160
entering receive state conditionally 222
entering receive state conditionally and including data 232
entering receive state unconditionally 251
entering receive state unconditionally and including data 243
flushing the buffer 445
immediate allocation request 43
indicating the beginning of a synchronization exchange 493
indicating the end of a synchronization exchange 497
obtain status on information available on a specified conversation 518
obtain status on information from any active conversation 502
obtain status on information immediately available from any active conversation 507
obtain status on information immediately available on a specified conversation 512
queued deallocation due to program, service, timer, or user errors 129
receive normal information immediately available from any conversation 328
receive normal information immediately available on a specific conversation 339
receiving an FMH-5 299
receiving expedited data from the specified conversation 280
receiving expedited data immediately available in any-mode 265
receiving expedited data immediately from the specified conversation 272
receiving expedited data in any-mode 258
request parameter list (RPL) (continued)
keyword, used with (continued)
rejecting a session 149
releasing a suspended session 484
replying to a confirmation request 391
requesting permission to enter send state 452
resetting the continuation mode 377
restoring a mode 166
sending an error notification 433
sending data 400
sending data and flushing the buffer 422
sending data conditionally 383, 411
sending expedited data on a full-duplex session 460
specific-mode RECEIVES 351
suspending a subsequent conversation 488
terminating a session 360
terminating a session due to errors 371
terminating a session due to protocol errors or cleanup deactivation 366
unconditional deallocation 121
unconditional deallocation that includes data 112
RESETRCV control value 372
responding negatively to session requests used with terminating a session 146
responding positively to session requests 132
RESTORE macroinstruction 162
RESTORE qualify value 162
restoring modes and sessions 162
return codes
listed and described 533
RCPRI, RCSEC combinations 533
RTNCD, FDB2 information 566
returned information
request-to-send indicator 448
RFC (request for comments)
accessing online xv
RPL (request parameter list) keyword, used with
accepting a session 136
allocation request for contention-winner session 33
allocation request for specific session 22
allocation request that can be queued 12
allocation request, conditional, without wait 54
any-mode RECEIVES 317
CNOS requests 142
conditional deallocation 92
conditional deallocation that includes data 102
CONTROL=CHECK 60
deallocation due to program errors 64
deallocation due to service errors 71
deallocation due to timer errors 78
deallocation due to user-specified criteria 85
defining LU-mode table values 154
displaying LU-mode table values 160
entering receive state conditionally 222
entering receive state conditionally and including data 232
entering receive state unconditionally 251
entering receive state unconditionally and including data 243
flushing the buffer 445
immediate allocation request 43

RPL (request parameter list) (continued)

keyword, used with (continued)
indicating the beginning of a synchronization exchange 493
indicating the end of a synchronization exchange 497
obtain status on information available on a specified conversation 518
obtain status on information from any active conversation 502
obtain status on information immediately available from any active conversation 507
obtain status on information immediately available on a specified conversation 512
queued deallocation due to program, service, timer, or user errors 129
receive normal information immediately available from any conversation 328
receive normal information immediately available on a specific conversation 339
receiving an FMH-5 299
receiving expedited data from the specified conversation 280
receiving expedited data immediately available in any-mode 265
receiving expedited data immediately from the specified conversation 272
receiving expedited data in any-mode 258
rejecting a session 149
releasing a suspended session 484
replying to a confirmation request 391
requesting permission to enter send state 452
resetting the continuation mode 377
restoring a mode 166
sending an error notification 433
sending data 400
sending data and flushing the buffer 422
sending data conditionally 383, 411
sending expedited data on a full-duplex session 460
specific-mode RECEIVES 351
suspending a subsequent conversation 488
terminating a session 360
terminating a session due to errors 371
terminating a session due to protocol errors or cleanup deactivation 366
unconditional deallocation 121
unconditional deallocation that includes data 112

RTNCD field (continued)
returned value, used with (continued)
deallocation due to timer errors 79
deallocation due to user-specified criteria 87
defining LU-mode table values 155
displaying LU-mode table values 161
entering receive state conditionally 224
entering receive state conditionally and including data 234
entering receive state unconditionally 252
entering receive state unconditionally and including data 245
flushing the buffer 446
immediate allocation request 46
indicating the beginning of a synchronization exchange 494
indicating the end of a synchronization exchange 498
obtain status on information available on a specified conversation 519
obtain status on information from any active conversation 503
obtain status on information immediately available from any active conversation 507
obtain status on information immediately available on a specified conversation 513
queued deallocation due to program, service, timer, or user errors 130
receive normal information immediately available from any conversation 331
receive normal information immediately available on a specific conversation 342
receiving an FMH-5 302
receiving expedited data from the specified conversation 282
receiving expedited data immediately available in any-mode 267
receiving expedited data immediately from the specified conversation 274
receiving expedited data in any-mode 259
rejecting a session 149
releasing a suspended session 484
replying to a confirmation request 392
requesting permission to enter send state 453
resetting the continuation mode 378
restoring a mode 167
sending an error notification 436
sending data 402
sending data and flushing the buffer 425
sending data conditionally 385, 413
sending expedited data on full-duplex session 462
specific-mode RECEIVES 353
suspending a subsequent conversation 489
terminating a session 361
terminating a session due to errors 372
terminating a session due to protocol errors or cleanup deactivation 366
unconditional deallocation 123
unconditional deallocation that includes data 114

RTSRTRN keyword, used with
allocation request for contention-winner session 33
allocation request for specific session 22
allocation request that can be queued 12
allocation request, conditional, without wait 54
RTSRTRN keyword, used with (continued)
immediate allocation request 43
ISTRPL6 530
receiving an FMH-5 299

S

security
conversation level 576
sending data
APPCCMD macroinstructions used
syntax and operands 379, 448
buffer list (OPTCD=BUFLST) considerations
conditional deallocation that includes data 100
entering receive state conditionally and including data 231
sending data 101, 399
sending data and flushing the buffer 231, 242, 421
sending data conditionally 410
sending expedited data on a full-duplex session 460
unconditional deallocation that includes data 112
flushing the buffer, used with
conditional deallocation 88
conditional deallocation that includes data 96
entering receive state conditionally and including data 226
entering receive state unconditionally 247
entering receive state unconditionally and including data 237
flushing the buffer 441
sending data and flushing the buffer 416
sending data conditionally 379, 405
unconditional deallocation 117
unconditional deallocation that includes data 107
macroinstructions for sending new data, used with
conditional deallocation that includes data 96
entering receive state conditionally and including data 226
entering receive state unconditionally and including data 237
sending data 394
sending data and flushing the buffer 416
sending data conditionally 405
unconditional deallocation 117
unconditional deallocation that includes data 107
sending expedited data on a full-duplex session 454
SENSE (continued)
returned value, used with (continued)
conditional deallocation 94
conditional deallocation that includes data 104
entering receive state conditionally 224
entering receive state conditionally and including data 234
entering receive state unconditionally and including data 245
receive normal information immediately available from any conversation 331
receive normal information immediately available on a specific conversation 342
receiving an FMH-5 302
sending an error notification 436
sending data 402
sending data and flushing the buffer 425
sending data conditionally 385, 413
specific-mode RECEIVEs 354
unconditional deallocation that includes data 114
SESSID
keyword, used with
indicating the beginning of a synchronization exchange 493
indicating the end of a synchronization exchange 497
terminating a session 371
returned value, used with allocation request for contention-winner session 36
allocation request for specific session 25
allocation request that can be queued 15
allocation request, conditional, without wait 57
immediate allocation request 46
receiving an FMH-5 302
releasing a suspended session 484
suspending a subsequent conversation 488
SESSIDL
keyword, used with
indicating the beginning of a synchronization exchange 493
indicating the end of a synchronization exchange 498
releasing a suspended session 484
terminating a session due to errors 371
returned value, used with allocation request for contention-winner session 36
allocation request for specific session 25
allocation request that can be queued 15
allocation request, conditional, without wait 57
immediate allocation request 46
receiving an FMH-5 302
releasing a suspended session 484
suspending a subsequent conversation 489
session activation, macroinstruction for 132
session deactivation
used with terminating a session macroinstruction 146
session limit
changing with CNOS requests 137
session limits, changing
with CNOS request macroinstruction 137
session parameters.
DSECT 569
relationship to RESTORE 162
session requests, accepting
session requests, accepting (continued)
  with a session macroinstruction 132
session requests, rejecting
  used with terminating a session 146
SETSESS control value 481, 485, 490, 494
shortcut keys 615
SIGDATA RPL extension field
  returned value, used with
    any-mode RECEIVES 320
    conditional deallocation that includes data 104
    entering receive state conditionally and including data 234
    entering receive state unconditionally and including data 245
    receive normal information immediately available on a specific conversation 342
    receiving expedited data from the specified conversation 282
    receiving expedited data immediately available in any-mode 267
    receiving expedited data immediately from the specified conversation 274
    receiving expedited data in any-mode 259
    sending an error notification 436
    sending data 403
    sending data and flushing the buffer 425
    sending data conditionally 413
    sending data conditionally 414
  unconditional deallocation that includes data 114
Signal RU 448
SIGRCV RPL extension field
  returned value, used with
    any-mode RECEIVES 320
    conditional deallocation that includes data 104
    entering receive state conditionally and including data 234
    entering receive state unconditionally and including data 245
    receive normal information immediately available from any conversation 331
    receive normal information immediately available on a specific conversation 342
    receiving expedited data from the specified conversation 282
    receiving expedited data immediately available in any-mode 267
    receiving expedited data immediately from the specified conversation 274
    receiving expedited data in any-mode 259
    sending an error notification 436
    sending data 403
    sending data and flushing the buffer 425
    sending data conditionally 413
    unconditional deallocation that includes data 115
STSHBF
  returned value, used with
    conditional deallocation that includes data 104
    deallocation due to program errors 66
    deallocation due to service errors 73
    deallocation due to timer errors 79
    deallocation due to user-specified errors 87
    entering receive state conditionally and including data 235
    entering receive state unconditionally and including data 245
    queued deallocation due to program, service, timer, or user errors 130
    sending an error notification 436
    sending data 403
    sending data and flushing the buffer 425
    sending data conditionally 414
    unconditional deallocation that includes data 115
STSHDS
  returned value, used with
    conditional deallocation that includes data 105
    deallocation due to service errors 73
    deallocation due to timer errors 80
    deallocation due to user-specified errors 87
    entering receive state conditionally and including data 235
    entering receive state unconditionally and including data 245
    queued deallocation due to program, service, timer, or user errors 131
    sending an error notification 437
    sending data 403
    sending data and flushing the buffer 425
    sending data conditionally 414
    unconditional deallocation that includes data 115
summary of changes xix
Summary of changes xix
sync level 577
T
TCP/IP
  online information xv
Tech notes xiii
terminating session
  terminating a session due to errors (syntax and operands) 367
trademark information 622
truncating
  sending an error notification macroinstruction 427
TYPE keyword, used with
  ISTRPL6 531
  sending an error notification 433
USERFLD, used with

- accepting a session 136
- allocation request for contention-winner session 34
- allocation request for specific session 22
- allocation request that can be queued 12
- allocation request, conditional, without wait 54
- CNOS request 142
- defining LU-mode table values 154
- displaying LU-mode table values 160
- immediate allocation request 44
- ISTRPL6 531
- receiving an FMH-5 300
- rejecting a session 149
- restoring a mode 166

returned value, used with

- accepting a session 136
- any-mode RECEIVES 321
- CNOS request 144
- conditional deallocation 94
- conditional deallocation that includes data 105
- deallocation due to program errors 66
- deallocation due to service errors 73
- deallocation due to timer errors 80
- deallocation due to user-specified criteria 87
- defining LU-mode table values 155
- displaying LU-mode table values 161
- entering receive state conditionally 224
- entering receive state conditionally and including data 235
- entering receive state unconditionally 252
- entering receive state unconditionally and including data 246
- flushing the buffer 446
- obtain status on information available on a specified conversation 519
- obtain status on information immediately available on a specified conversation 513
- queued deallocation due to program, service, timer, or user errors 131
- receive normal information immediately available from any conversation 331
- receive normal information immediately available on a specific conversation 343
- receiving expedited data from the specified conversation 282
- receiving expedited data immediately available in any-mode 267
- receiving expedited data immediately from the specified conversation 275
- receiving expedited data in any-mode 260
- rejecting a session 149
- replying to a confirmation request 392
- requesting permission to enter send state 453
- resetting the continuation mode 378
- restoring a mode 167
- sending an error notification 437
- sending data 403
- sending data and flushing the buffer 425
- sending data conditionally 386, 414
- sending expedited data on full-duplex session 462
- specific-mode RECEIVES 354

VTAM, online information xv

what-received RPL extension field

returned value, used with

- any-mode RECEIVES 321, 331
- receive normal information immediately available from any conversation 331
- receive normal information immediately available on a specific conversation 343
- specific-mode RECEIVES 354

WHENFREE qualify value

- session assignment algorithm 48
- winner, contention requests for allocation 27

z/OS Basic Skills Information Center xiii
z/OS, documentation library listing 623
Communicating your comments to IBM

If you especially like or dislike anything about this document, you can send us comments electronically by using one of the following methods:

**Internet email:**
comsvrcf@us.ibm.com

**World Wide Web:**

If you would like a reply, be sure to include your name, address, and telephone number. Make sure to include the following information in your comment or note:

- Title and order number of this document
- Page number or topic related to your comment

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this document. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.