

## **System Takeover Facility**

### **Background**

A fundamental characteristic of ALCS is very high availability. This is a characteristic that is continually focused on and has been subject to some recent developments.

### **Sysplex**

To achieve 100% availability consideration has been given to implementing a Sysplex ALCS were multiple ALCS regions appear as one system to the end user. However having researched and discussed a Sysplex ALCS with customers the conclusion is that the cost to develop, implement and run a system is not affordable at this time.

### **Faster restart**

Without going to the lengths of a Sysplex ALCS availability can be enhanced by reducing outages and shortening the duration of outages. To reduce outage duration in ALCS new functions have:

1. Increased the speed of ALCS diagnostic dump processing. However using MVS SVC dump for catastrophic errors is far quicker
2. Shortened the IDLE to HALT state delay loops
3. Made general file open processing parallel rather than sequential

A facility available with ALCS is the system takeover facility. With this a variety of standby system, the automatic standby, can takeover the database and restart once the primary system has released the database. The benefits are a faster system restart with no operator intervention

### **Standby System**

Before describing the automatic standby recall the standby ALCS function. A standby ALCS is started using an ALCS system state parameter of STANDBY. The standby system performs its pre-initialization. This involves allocating storage, loading the configuration tables and building work areas and control blocks. Once this work is complete it issues a message to the operator:

DXC110R Standby state – Reply with required system state (IDLE, CRAS, MESW, or NORM) or CANCEL

It then is swapped out of storage and uses little system resource, until the operator replies to the above message to perform a restart or to cancel the system.

This reduces the time required to start but depends on operator intervention. Often it is used to reduce the time of scheduled outages.

## **Automatic Standby**

An ALCS automatic standby system is started using a new set of ALCS system state parameters prefixed with the letter X. The new states are "XI", or "XIDLE", or "XC", or "XCRAS", or "XM", or "XMESW", or "XN" or "XNORM".

Only one automatic standby ALCS is allowed at any one time. If a second ALCS automatic standby system is started it will fail with message

DXC117T Initialization failed -- Another ALCS takeover is waiting

It is possible to have 'non-automatic' ALCS standby systems concurrently, but be aware that such a scenario could result in extra operational complexity.

The automatic standby ALCS system differs from a standby ALCS system in that it is not dormant. An automatic standby does not issue a DXC110R message. Instead it requests to enqueue the ALCS database and assuming it fails it issues:

DXC116D Can not obtain exclusive control for database – ALCS takeover is now waiting -- Reply C to cancel takeover

The only reply possible to this message is to cancel the automatic standby system. The message is removed if the system does takeover.

The automatic standby system continues in a partially initialized state. It is non-swappable and active using system resources, although it uses only a little CPU resource. It retries every second to request the database enqueue. Once the enqueue is available the system will complete restart.

## **Automatic standby system in the same LPAR**

This is straightforward to implement, however the circumstances where it protects against an outage are limited. As the automatic standby system is sharing the host hardware and software as the active system any failure of the hardware, operating system software or additional service software such as VTAM, TCP/IP, MQ will have similar effect on both active and automatic standby systems.

Take care that there is sufficient REAL storage as the alternate system will not be swapped out. Recent machines have far greater real storage specifications and can exceed 2 gigabytes. With the current versions of ALCS, VFA buffers to be backed anywhere in real storage thus reducing the usage of real storage below the BAR (2 Gigabyte boundary).

## **Automatic standby system in a different LPAR**

When the ALCS automatic standby runs on a different host, both the automatic standby host and the live ALCS system host must belong to the same GRS complex. This is essential so that the enqueue on the ALCS database can be requested and held by either system.

It is advisable that both systems belong to the same SYSPLEX as GRS in a SYSPLEX is much more robust and easier to operate compared with GRS enabled through channel to channel connectors.

The automatic standby system requires access to the ALCS DASD, TAPES and Communication Front End Processors.

## **Enqueue Considerations**

The enqueue on the database is an operating system 'ENQ' request from ALCS to reserve a resource of majorname 'DXCMON' with minorname the name of the ALCS DASD configuration table as specified in the EXEC parameter card.

Whilst ALCS is running the enqueue is for a shared resource. When ALCS is attempting to start it requests exclusive control of the resource to ensure another ALCS is not active.

It is important that you are consistent with the use of the name of the DASD configuration in the EXEC card as this is the resource that is being reserved.

## **Network Considerations**

Setting up ALCS to switch to an alternate LPAR is only part of the action required. To have the system available to end users consideration needs to be given to switching the network components. Procedures may currently be in place to perform these actions already. There are many approaches that could be taken to automate these actions and any method will be dependent upon systems, and facilities available locally. Software for automation and system management, such as Tivoli NetView, if present should be considered.

To assist with automation specific enhancements have been made to ALCS. Messages were added indicating when the VTAM ACB has been opened and closed. Virtual IP addressing support was added.

Two parts Communication Server to consider are VTAM and TCPIP.

## **VTAM ACB**

The ALCS VTAM ACB(s) must be switched to the alternate system. This involves inactivating the VTAM ACB on the former system and moving it to the new system. To enable this the ALCS VTAM ACBs need to be defined in a VTAM application major node on both systems. This VTAM application major node must only contain the ALCS VTAM ACBs. Make use of ADJSSCP selection. This is the preferred way to reduce or eliminate cross-domain LU definitions.

To enable automation ALCS issues the following messages:

```
DXC393I VTAM ACB LUN-'acbname' open failed, Return code RC-X'return code' -- ALCS takeover will retry
```

This will appear at restart when the automatic standby takes over and requires that the ACB is active. The newly active ALCS will retry once to allow for automation to take action. If it is still unsuccessful message DXC200R will be sent.

```
DXC394I VTAM ACB LUN-'acbname' has been closed
```

This will appear when ALCS ends contact with VTAM during termination either normally (ZASYS HALT) or abnormally.

```
DXC395I VTAM ACB LUN-'acbname' has been opened
```

This will appear when ALCS contacts VTAM during initialization.

## Virtual IP addressing (VIPA)

Virtual IP addressing (VIPA) makes TCP/IP applications independent of physical Internal Protocol (IP) addresses, improving the ease of system management. ALCS supports VIPA with an optional parameter, TCPVIPA, on the SCTGEN.

ALCS normally accepts incoming requests over all available network interfaces. ALCS issues BIND(INADDR\_ANY) when it creates TCP/IP sockets for use by the concurrent server (Listener) and by TCP/IP communication resources.

ALCS can use BIND(specified IP Address) which can be exploited when ALCS is restarted on a different LPAR or different machine. If the LPAR that ALCS resides in fails a dynamic VIPA can be specified to move to a backup stack in a SYSPLEX. If ALCS or another component fails on the live system and the operators want to restart ALCS elsewhere then they can move the VIPA to another IP stack in a different LPAR, using VARY OBEY commands on the backup system.

The moving of a ALCS VIPA could be automated, the event messages to initiate this can be the ALCS VTAM ACB open / close messages described in the VTAM ACB section.

In all cases current connections are lost but end users will connect on next input with the restarted ALCS.

For more information on moving a VIPA refer to the z/OS Communication Server IP Configuration Guide.

## WTO Exits

For installations without systems management software, message exits can be used to automate the switching of the network components. The manual z/OS MVS Installation Exits contains a section IEAVMXIT – Installation-Specified MPF Exits describing user message processing.

Sample exits for the switching of the ALCS VTAM ACB are described here. These are provided on an as is basis and require review and tailoring for implementation. The exits open an ACB for a VTAM secondary program operator that enables commands to be sent to VTAM.

The exits could be enhanced to use the MGCRC macro to submit MVS commands. They could invoke the necessary commands to switch the virtual IP address. Information on the MVS MGCRC macro can be found in the MVS Authorised Assembler Programming Guide and Reference and after the sample exits at the end of this document.

### Restriction:

In the event of a CANCEL of ALCS the DXC394I VTAM ACB closed message is issued but the sample message exit will fail. ALCS is being cancelling and will not be permitted to open an ACB. In this case operator intervention will be required.

## DXC393I WTO sample exit

Please **read** the comments in the WTO sample exit. The exit requires definition of a VTAM secondary program operator (SPO) ACB in the VTAMLST. (In the exit the ACBNAME is NETVSACB at statement APPLID ....). The exit also has naming dependencies on the VTAM major node ALCS ACB definition.

```
TITLE 'DXC393I WTO USER EXIT FOR ALCS'
PRINT NOGEN

*-----*
* DXC393I WTO MESSAGE EXIT ROUTINE *
*-----*
* THIS ROUTINE GETS CONTROL WHEN ALCS ISSUES A DXC393I MESSAGE: *
* DXC393I COM M VTAM ACB LUN-'MALCSACB' *
* *
* TO ACTIVATE THIS ROUTINE: *
* 1) ENTER THE FOLLOWING RECORD IN SYS1.PARMLIB(MPFLSTXX): *
* DXC393I,SUP(NO),USEREXIT(DXC393I) *
* 2) LINKEDIT THIS ROUTINE INTO ANY LNKLST LIBRARY *
* 3) REFRESH THE LLA BY ENTERING: *
* F LLA,REFRESH *
* 4) AT THE MVS CONSOLE, ENTER: *
* SET MPF=XX *
* (XX IS THE TWO CHARACTER SUFFIX IN MPFLSTXX) *
*-----*
* ENTRY CONDITIONS: *
* R1 - ADDRESS OF CTXT (PARAMETER LIST, MAPPED BY IEZVX100) *
* R13 - SAVE AREA *
* R14 - RETURN ADDRESS *
* R15 - ENTRY POINT ADDRESS *
*-----*
* EXAMPLE OF SPO ACBNAME: *
* VBUILD TYPE=APPL *
* NETVSACB APPL AUTH=(SPO,ACQ),ACBNAME=NETVSACB,SRBEXIT=YES, *
* DLOGMOD=ALCSPARS *
* *
* EXAMPLE OF ALCS ACBNAME: (ONLY ACB IN THE MAJOR GROUP GALCSACB) *
* MALCSACB APPL AUTH=(ACQ),ACBNAME=MALCSACB,SRBEXIT=YES, *
* PARSESS=YES,DLOGMOD=ALCSPARS *
* STATOPT=('ALCS APPL ') *
* *
* THIS VERSION REQUIRES THAT THE ACBNAME IS EIGHT CHARACTERS. MAJOR *
* GROUPNAME MUST START WITH G. (FOR EXAMPLE: ACBNAME MALCSACB *
* REQUIRES MAJOR GROUP NAME GALCSACB) *
*-----*
EJECT ,

*-----*
* EXIT CONDITIONS: *
* ALL REGISTERS RESTORED *
* *
* NB. U2001 INDICATES AN ERROR, IF R07 CONTAINS A RETURN CODE. *
* SEE THIS PROGRAM FOR DETAILS. *
*-----*
EJECT ,
```

```

DXC393I  CSECT
DXC393I  AMODE 31
DXC393I  RMODE 24
        SPACE 1
***** REGISTER EQUATES
        SPACE 1
R00      EQU 0
R01      EQU 1
R02      EQU 2
R03      EQU 3          MESSAGE
R04      EQU 4          MESSAGE TEXT
R05      EQU 5          CTXT
R06      EQU 6          ADDRESS OF ACB NAME
R07      EQU 7          ERROR CODE OR ZEROES
R08      EQU 8          ADDRESS OF ACB
R09      EQU 9          ADDRESS OF RPL
R10      EQU 10
R11      EQU 11         GETMAINED DATA AREA BASE
R12      EQU 12         BASE REGISTER
R13      EQU 13         SAVE AREA
R14      EQU 14         RETURN ADDRESS
R15      EQU 15         ENTRY POINT ADDRESS
        EJECT ,
***** STANDARD ENTRY LINKAGE
        SPACE 1
STM      R14,R12,12(R13)  SAVE REGISTERS
LR       R12,R15          LOAD BASE REGISTER
USING   DXC393I,R12
L        R05,0(,R01)     ADDRESS CTXT
USING   CTXT,R05
        SPACE 1
***** OBTAIN STORAGE
        SPACE 1
GETMAIN RC,LV=DATAEND,SP=230 OBTAIN STORAGE
LTR      R15,R15          STORAGE OBTAINED
BNZ     RETURN3          BRANCH IF NOT - RETURN
        SPACE 1
LR       R11,R01          ADDRESS WORKING STORAGE
USING   DATAAREA,R11
ST       R13,SAVE+4      STANDARD SAVEAREA CHAINING
LA       R15,SAVE
ST       R15,8(,R13)
LR       R13,R15
        SPACE 1
***** SET ESTAE
        SPACE 1
MVC     ESTEAL(ESTEASL),ESTEAS COPY ESTAE MODEL
STM     R12,R13,ESTEAP   SAVE REGISTERS IN ESTAE PARM AREA
LA      R04,ESTEAP      LOAD ADDRESS OF PARM AREA
ST      R04,ESTEAPL     STORE IN ADDRESS IN ESTEA PARM LIST
OI      ESTEAPL,X'80'   INDICATE THIS IS LAST IN PARM LIST
        SPACE 1
LA      R03,EXIT        LOAD ADDRESS OF ESTAE EXIT
LA      R04,ESTEAPL     LOAD ADDRESS OF ESTAE PARM LIST

```

```

SPACE 1
ESTAE (R03),PARAM=(R04),MF=(E,ESTEAL) SET ESTAE
SPACE 1
***** ACCESS MESSAGE TEXT
SPACE 1
L R03,CTXTXPJ ADDRESS MESSAGE
USING CTXTATTR,R03
LA R04,CTXTMSG ADDRESS MESSAGE TEXT
USING MSGTEXT,R04
EJECT ,
***** PROCESS MESSAGE
SPACE 1
LA R08,VTAMACB ADDRESS OF ACB
LA R06,APPLID ADDRESS OF ACB NAME
SPACE 1
GENCB BLK=ACB,AM=VTAM,WAREA=(R08),LENGTH=256,APPLID=(R06), *
MF=(G,GENA,GENB)
SPACE 1
LTR R15,R15 GENCB OK
BNZ GENF1 BRANCH IF NOT
SPACE 1
MVC OPENL(OPENSL),OPENS COPY SKELETON
SPACE 1
OPEN ((R08)),MF=(E,OPENL) OPEN ACB
SPACE 1
LTR R07,R15 OPEN OK
BNZ OPENFD BRANCH IF NOT
SPACE 1
LA R09,SENRPL ADDRESS OF RPL
USING IFGRPL,R09
SPACE 1
GENCB BLK=RPL,AM=VTAM,WAREA=(R09),LENGTH=256,MF=(G,GENC,GEND)
SPACE 1
LTR R15,R15 GENCB OK
BNZ GENF2 BRANCH IF NOT
SPACE 1
MVC COMMAND(CMDLN),SKELETON COPY SKELETON TO COMMAND
MVC COMMAND+13(L'MSGACB),MSGACB COPY ACB NAME
MVI COMMAND+13,C'G' MAKE IT GROUPNAME
SPACE 1
SENDCMD RPL=(R09),AREA=COMMAND,RECLN=CMDLN,ACB=(R08)
SPACE 1
LTR R15,R15 SEND OK
BZ RETURNOK BRANCH IF YES
EJECT ,
***** ERROR ROUTINES
SPACE 1
SENDFD DC 0H'0'
MVC CLOSEL(CLOSESL),CLOSES COPY SKELETON
CLOSE ((R08)),MF=(E,CLOSEL) CLOSE ACB
SPACE 1
SR R07,R07 SET TO ZEROES
ICM R07,B'1000',RPLRTNCD RPL RETURN CODE
ICM R07,B'0100',RPLFDB2 RPL FDB2 STATUS

```

```

        ICM  R07,B'0001',,=AL1(16) SENDCMD FAILED
        B    RETURN1          RETURN
        SPACE 1
OPENFD  DC    0H'0'
        SLL  R07,8           SHIFT ONE BYTE
        ICM  R07,B'0001',,=AL1(20) OPEN ACB FAILED
        B    RETURN1          RETURN
        SPACE 1
GENF1   DC    0H'0'
        LA   R07,8           ACB GENCB FAILED
        B    RETURN1          RETURN
        SPACE 1
GENF2   DC    0H'0'
        MVC  CLOSEL(CLOSESL),CLOSES COPY SKELETON
        CLOSE ((R08)),MF=(E,CLOSEL) CLOSE ACB
        SPACE 1
        LA   R07,12          RPL GENCB FAILED
        B    RETURN1          RETURN
        EJECT ,
***** STANDARD EXIT LINKAGE
        SPACE 1
RETURNOK DC    0H'0'
        MVC  CLOSEL(CLOSESL),CLOSES COPY SKELETON
        CLOSE ((R08)),MF=(E,CLOSEL) CLOSE ACB
        SPACE 1
        SR   R07,R07          SET TO ZEROES
*       B    RETURN1          CONTINUE
        SPACE 1
RETURN1  DC    0H'0'
        LTR  R07,R07          ANY ERROR
        BZ   RETURN2          BRANCH IF NOT
        SPACE 1
        ABEND 2001,DUMP,STEP,USER ABEND
        SPACE 1
RETURN2  DC    0H'0'
        ESTAE 0,MF=(E,ESTEAL)  CANCEL ESTAE
        SPACE 1
        L    R13,SAVE+4       LOAD ADDRESS OF SAVE AREA
        FREEMAIN RU,LV=DATAEND,A=(R11),SP=230 FREEMAIN WORKAREA
        SPACE 1
RETURN3  DC    0H'0'
        LM   R14,R12,12(R13)  RESTORE SAVED REGISTERS R14-R12
        BR   R14              RETURN
        EJECT ,
***** DATA AREAS
        DS   0D
APPLID   DC    AL1(8),C'NETVSACB'
        SPACE 1
SKELETON DC    X'00',X'01',XL2'00'
        DC   CL80'V NET,ID=MALCSACB,ACT'
SKELLN   EQU   *-SKELETON
        SPACE 1
CLOSES   CLOSE (),MF=L       CLOSE ACB
CLOSESL  EQU   *-CLOSES

```

```

OPENS      OPEN  ( ),MF=L              OPEN  ACB
OPENSL     EQU   *-OPENS
           SPACE 1
ESTEAS     ESTAE ESTEAS,PARAM=ESTEAS,MF=L
ESTEASL    EQU   *-ESTEAS
           SPACE 1
           LTORG
           EJECT ,
*****    ESTAE RETRY ROUTINE
           SPACE 1
RETRY      DC    0H'0'
           L     R01,0(,R01)           LOAD ADDRESS OF PARAMETERS
           LM    R12,R13,0(R01)       RESTORE REGISTERS R12 AND R13
                                           (BASE AND SAVE AREA ADDRESS)
           B     RETURN2              CANCEL ESTAE AND RETURN
           SPACE 1
*****    ESTAE EXIT
           SPACE 1
EXIT       DC    0H'0'
           DROP  R12                  DROP BASE
           USING *,R15                USE REGISTER R15 FOR THIS EXIT
           SPACE 1
           CH    R00,=Y(12)          SDWA PROVIDED
           BNE   EXIT1               BRANCH IF YES
           SPACE 1
           LA    R15,0                PERCOLATE ABEND
           BR    R14                  RETURN TO RTM
           SPACE 1
EXIT1      DC    0H'0'
           STM   R14,R12,12(R13)     SAVE REGISTERS (DO NOT PROVIDE
                                           A NEW SAVE AREA)
           SPACE 1
           L     R10,=A(RETRY)        ADDRESS OF RETRY ROUTINE
           SPACE 1
           SETRP RC=4,RETREGS=NO,RETADDR=(10),FRESWA=YES,RECORD=YES,
           REGS=(14,12),DUMP=YES RETRY AND RETURN
           SPACE 1
           LTORG ,
           EJECT ,
DATAAREA   DSECT
SAVE       DS    18F
           SPACE 1
           DC    CL8'ACB'
VTAMACB   DC    XL256'00'
           DC    XL100'00'
           SPACE 1
           DC    CL8'RPL'
SENRPL    DC    XL256'00'
           DC    XL100'00'
           SPACE 1
GENA      DC    (GENB)X'00'
GENC      DC    (GEND)X'00'
           SPACE 1
CLOSEL    CLOSE ( ),MF=L            CLOSE ACB

```

|         |          |                     |        |                         |
|---------|----------|---------------------|--------|-------------------------|
| OPENL   | OPEN     | ( ),MF=L            | OPEN   | ACB                     |
|         | SPACE    | 1                   |        |                         |
| ESTEAL  | DS       | CL(ESTEASL)         | ESTAE  |                         |
|         | SPACE    | 1                   |        |                         |
| ESTEAPL | DS       | F                   | ESTEAP | ESTEA PARM LIST         |
| ESTEAP  | DS       | 2F                  | ESTEAP | ESTEA PARAMETERS        |
|         | SPACE    | 1                   |        |                         |
| COMMAND | DC       | X'00',X'01',XL2'00' |        |                         |
|         | DS       | CL80                |        |                         |
| CMDLN   | EQU      | *-COMMAND           |        |                         |
| DATAEND | EQU      | *-DATAAREA          |        |                         |
| MSGTEXT | DSECT    | ,                   |        | MAPPING OF MESSAGE TEXT |
| MSGID   | DS       | CL8                 |        | MESSAGE ID 'DXC393I'    |
| MSGCOMP | DS       | CL3                 |        | COMPONENT 'COM'         |
|         | DS       | C                   |        |                         |
| MSGALID | DS       | C                   |        | ALCS ID                 |
|         | DS       | C                   |        |                         |
|         | DC       | C' VTAM ACB LUN-''' |        |                         |
| MSGACB  | DC       | CL8'MALCSACB'       |        |                         |
|         | SPACE    | 1                   |        |                         |
|         | IEZVX100 | ,                   |        | CTXT MAPPING            |
|         | SPACE    | 1                   |        |                         |
|         | IFGRPL   | AM=VTAM             |        | VTAM RPL                |
|         | SPACE    | 1                   |        |                         |
|         | IHASDWA  | DSECT=YES           |        | SDWA                    |
|         | END      | DXC393I             |        |                         |

## DXC394I WTO sample exit

Please **read** the comments in the WTO sample exit. The exit requires definition of a VTAM secondary program operator (SPO) ACB in the VTAMLST. (In the exit the ACBNAME is NETVSACB at statement APPLID ....). The exit also has naming dependencies on the VTAM major node ALCS ACB definition.

```
TITLE 'DXC394I WTO USER EXIT FOR ALCS'
PRINT NOGEN
*-----*
* DXC394I WTO MESSAGE EXIT ROUTINE *
*-----*
* THIS ROUTINE GETS CONTROL WHEN ALCS ISSUES A DXC394I MESSAGE: *
* DXC394I COM M VTAM ACB LUN-'MALCSACB' *
* *
* TO ACTIVATE THIS ROUTINE: *
* 1) ENTER THE FOLLOWING RECORD IN SYS1.PARMLIB(MPFLSTXX): *
* DXC394I,SUP(NO),USEREXIT(DXC394I) *
* 2) LINKEDIT THIS ROUTINE INTO ANY LNKLST LIBRARY *
* 3) REFRESH THE LLA BY ENTERING: *
* F LLA,REFRESH *
* 4) AT THE MVS CONSOLE, ENTER: *
* SET MPF=XX *
* (XX IS THE TWO CHARACTER SUFFIX IN MPFLSTXX) *
*-----*
* ENTRY CONDITIONS: *
* R1 - ADDRESS OF CTXT (PARAMETER LIST, MAPPED BY IEZVX100) *
* R13 - SAVE AREA *
* R14 - RETURN ADDRESS *
* R15 - ENTRY POINT ADDRESS *
*-----*
* *
* EXAMPLE OF SPO ACBNAME: *
* VBUILD TYPE=APPL *
* NETVSACB APPL AUTH=(SPO,ACQ),ACBNAME=NETVSACB,SRBEXIT=YES, *
* DLOGMOD=ALCSPARS *
* *
* EXAMPLE OF ALCS ACBNAME: (ONLY ACB IN THE MAJOR GROUP GALCSACB) *
* MALCSACB APPL AUTH=(ACQ),ACBNAME=MALCSACB,SRBEXIT=YES, *
* PARSESS=YES,DLOGMOD=ALCSPARS *
* STATOPT=('ALCS APPL ') *
* *
* THIS VERSION REQUIRES THAT THE ACBNAME IS EIGHT CHARACTERS. MAJOR *
* GROUPNAME MUST START WITH G. (FOR EXAMPLE: ACBNAME MALCSACB *
* REQUIRES MAJOR GROUP NAME GALCSACB) *
*-----*
EJECT ,
*-----*
* EXIT CONDITIONS: *
* ALL REGISTERS RESTORED *
* *
* NB. U2001 INDICATES AN ERROR, IF R07 CONTAINS A RETURN CODE. *
* SEE THIS PROGRAM FOR DETAILS. *
*-----*
```

```

EJECT ,
DXC394I CSECT
DXC394I AMODE 31
DXC394I RMODE 24
SPACE 1
***** REGISTER EQUATES
SPACE 1
R00 EQU 0
R01 EQU 1
R02 EQU 2
R03 EQU 3 MESSAGE
R04 EQU 4 MESSAGE TEXT
R05 EQU 5 CTXT
R06 EQU 6 ADDRESS OF ACB NAME
R07 EQU 7 ERROR CODE OR ZEROES
R08 EQU 8 ADDRESS OF ACB
R09 EQU 9 ADDRESS OF RPL
R10 EQU 10
R11 EQU 11 GETMAINED DATA AREA BASE
R12 EQU 12 BASE REGISTER
R13 EQU 13 SAVE AREA
R14 EQU 14 RETURN ADDRESS
R15 EQU 15 ENTRY POINT ADDRESS
EJECT ,
***** STANDARD ENTRY LINKAGE
SPACE 1
STM R14,R12,12(R13) SAVE REGISTERS
LR R12,R15 LOAD BASE REGISTER
USING DXC394I,R12
L R05,0(,R01) ADDRESS CTXT
USING CTXT,R05
SPACE 1
***** OBTAIN STORAGE
SPACE 1
GETMAIN RC,LV=DATAEND,SP=230 OBTAIN STORAGE
LTR R15,R15 STORAGE OBTAINED
BNZ RETURN3 BRANCH IF NOT - RETURN
SPACE 1
LR R11,R01 ADDRESS WORKING STORAGE
USING DATAAREA,R11
ST R13,SAVE+4 STANDARD SAVEAREA CHAINING
LA R15,SAVE
ST R15,8(,R13)
LR R13,R15
SPACE 1
***** SET ESTAE
SPACE 1
MVC ESTEAL(ESTEASL),ESTEAS COPY ESTAE MODEL
STM R12,R13,ESTEAP SAVE REGISTERS IN ESTAE PARM AREA
LA R04,ESTEAP LOAD ADDRESS OF PARM AREA
ST R04,ESTEAPL STORE IN ADDRESS IN ESTEA PARM LIST
OI ESTEAPL,X'80' INDICATE THIS IS LAST IN PARM LIST
SPACE 1
LA R03,EXIT LOAD ADDRESS OF ESTAE EXIT

```

```

LA      R04,ESTEAPL      LOAD ADDRESS OF ESTAE PARM LIST
SPACE 1
ESTAE  (R03),PARAM=(R04),MF=(E,ESTEAL) SET ESTAE
SPACE 1
***** ACCESS MESSAGE TEXT
SPACE 1
L       R03,CTXTTYPJ     ADDRESS MESSAGE
USING  CTXTATTR,R03
LA      R04,CTXTTMSG     ADDRESS MESSAGE TEXT
USING  MSGTEXT,R04
EJECT ,
***** PROCESS MESSAGE
SPACE 1
LA      R08,VTAMACB      ADDRESS OF ACB
LA      R06,APPLID       ADDRESS OF ACB NAME
SPACE 1
GENCB  BLK=ACB,AM=VTAM,WAREA=(R08),LENGTH=256,APPLID=(R06), *
MF=(G,GENA,GENB)
SPACE 1
LTR     R15,R15          GENCB OK
BNZ     GENF1            BRANCH IF NOT
SPACE 1
MVC    OPENL(OPENSL),OPENS COPY SKELETON
SPACE 1
OPEN   ((R08)),MF=(E,OPENL) OPEN ACB
SPACE 1
LTR     R07,R15          OPEN OK
BNZ     OPENFD           BRANCH IF NOT
SPACE 1
LA      R09,SENDRPL      ADDRESS OF RPL
USING  IFGRPL,R09
SPACE 1
GENCB  BLK=RPL,AM=VTAM,WAREA=(R09),LENGTH=256,MF=(G,GENC,GEND)
SPACE 1
LTR     R15,R15          GENCB OK
BNZ     GENF2            BRANCH IF NOT
SPACE 1
MVC    COMMAND(CMDLN),SKELETON COPY SKELETON TO COMMAND
MVC    COMMAND+13(L'MSGACB),MSGACB COPY ACB NAME
MVI    COMMAND+13,C'G'   MAKE IT GROUPNAME
SPACE 1
SENDCMD RPL=(R09),AREA=COMMAND,RECLEN=CMDLN,ACB=(R08)
SPACE 1
LTR     R15,R15          SEND OK
BZ      RETURNOK        BRANCH IF YES
EJECT ,
***** ERROR ROUTINES
SPACE 1
SENDFD DC 0H'0'
MVC    CLOSEL(CLOSESL),CLOSES COPY SKELETON
CLOSE  ((R08)),MF=(E,CLOSEL) CLOSE ACB
SPACE 1
SR      R07,R07          SET TO ZEROES
ICM    R07,B'1000',RPLRTNCD RPL RETURN CODE

```

```

      ICM  R07,B'0100',RPLFDB2 RPL FDB2 STATUS
      ICM  R07,B'0001',=AL1(16) SENDCMD FAILED
      B    RETURN1                RETURN
      SPACE 1
OPENFD  DC    0H'0'
      SLL  R07,8                   SHIFT ONE BYTE
      ICM  R07,B'0001',=AL1(20) OPEN ACB FAILED
      B    RETURN1                RETURN
      SPACE 1
GENF1   DC    0H'0'
      LA   R07,8                   ACB GENCB FAILED
      B    RETURN1                RETURN
      SPACE 1
GENF2   DC    0H'0'
      MVC  CLOSEL(CLOSESL),CLOSES COPY SKELETON
      CLOSE ((R08)),MF=(E,CLOSEL) CLOSE ACB
      SPACE 1
      LA   R07,12                  RPL GENCB FAILED
      B    RETURN1                RETURN
      EJECT ,
***** STANDARD EXIT LINKAGE
      SPACE 1
RETURNOK DC    0H'0'
      MVC  CLOSEL(CLOSESL),CLOSES COPY SKELETON
      CLOSE ((R08)),MF=(E,CLOSEL) CLOSE ACB
      SPACE 1
      SR   R07,R07                 SET TO ZEROES
*       B    RETURN1                CONTINUE
      SPACE 1
RETURN1  DC    0H'0'
      LTR  R07,R07                 ANY ERROR
      BZ   RETURN2                BRANCH IF NOT
      SPACE 1
      ABEND 2001,DUMP,STEP,USER ABEND
      SPACE 1
RETURN2  DC    0H'0'
      ESTAE 0,MF=(E,ESTEAL)        CANCEL ESTAE
      SPACE 1
      L     R13,SAVE+4             LOAD ADDRESS OF SAVE AREA
      FREEMAIN RU,LV=DATAEND,A=(R11),SP=230 FREEMAIN WORKAREA
      SPACE 1
RETURN3  DC    0H'0'
      LM   R14,R12,12(R13)        RESTORE SAVED REGISTERS R14-R12
      BR   R14                    RETURN
      EJECT ,
***** DATA AREAS
      DS   0D
APPLID   DC    AL1(8),C'NETVSACB'
      SPACE 1
SKELETON DC    X'00',X'01',XL2'00'
      DC   CL80'V NET,ID=MALCSACB,INACT,FORCE'
SKELLN   EQU   *-SKELETON
      SPACE 1
CLOSES   CLOSE (),MF=L           CLOSE ACB

```

```

CLOSESL EQU *-CLOSES
OPENS OPEN ( ),MF=L OPEN ACB
OPENS EQU *-OPENS
SPACE 1
ESTEAS ESTAE ESTEAS,PARAM=ESTEAS,MF=L
ESTEASL EQU *-ESTEAS
SPACE 1
LTORG
EJECT ,
***** ESTAE RETRY ROUTINE
SPACE 1
RETRY DC 0H'0'
L R01,0(,R01) LOAD ADDRESS OF PARAMETERS
LM R12,R13,0(R01) RESTORE REGISTERS R12 AND R13 -
(BASE AND SAVE AREA ADDRESS)
B RETURN2 CANCEL ESTAE AND RETURN
SPACE 1
***** ESTAE EXIT
SPACE 1
EXIT DC 0H'0'
DROP R12 DROP BASE
USING *,R15 USE REGISTER R15 FOR THIS EXIT
SPACE 1
CH R00,=Y(12) SDWA PROVIDED
BNE EXIT1 BRANCH IF YES
SPACE 1
LA R15,0 PERCOLATE ABEND
BR R14 RETURN TO RTM
SPACE 1
EXIT1 DC 0H'0'
STM R14,R12,12(R13) SAVE REGISTERS (DO NOT PROVIDE -
A NEW SAVE AREA)
SPACE 1
L R10,=A(RETRY) ADDRESS OF RETRY ROUTINE
SPACE 1
SETRP RC=4,RETREGS=NO,RETADDR=(10),FRESWA=YES,RECORD=YES, -
REGS=(14,12),DUMP=YES RETRY AND RETURN
SPACE 1
LTORG ,
EJECT ,
DATAAREA DSECT
SAVE DS 18F
SPACE 1
DC CL8'ACB'
VTAMACB DC XL256'00'
DC XL100'00'
SPACE 1
DC CL8'RPL'
SENDRPL DC XL256'00'
DC XL100'00'
SPACE 1
GENA DC (GENB)X'00'
GENC DC (GEND)X'00'
SPACE 1

```

|         |                        |                         |
|---------|------------------------|-------------------------|
| CLOSEL  | CLOSE ( ),MF=L         | CLOSE ACB               |
| OPENL   | OPEN ( ),MF=L          | OPEN ACB                |
|         | SPACE 1                |                         |
| ESTEAL  | DS CL(ESTEASL)         | ESTAE                   |
|         | SPACE 1                |                         |
| ESTEAPL | DS F                   | ESTEAP PARM LIST        |
| ESTEAP  | DS 2F                  | ESTEAP PARAMETERS       |
|         | SPACE 1                |                         |
| COMMAND | DC X'00',X'01',XL2'00' |                         |
|         | DS CL80                |                         |
| CMDLN   | EQU *-COMMAND          |                         |
| DATAEND | EQU *-DATAAREA         |                         |
| MSGTEXT | DSECT ,                | MAPPING OF MESSAGE TEXT |
| MSGID   | DS CL8                 | MESSAGE ID 'DXC394I '   |
| MSGCOMP | DS CL3                 | COMPONENT 'COM'         |
|         | DS C                   |                         |
| MSGALID | DS C                   | ALCS ID                 |
|         | DS C                   |                         |
|         | DC C' VTAM ACB LUN-''' |                         |
| MSGACB  | DC CL8'MALCSACB'       |                         |
|         | SPACE 1                |                         |
|         | IEZVX100 ,             | CTXT MAPPING            |
|         | SPACE 1                |                         |
|         | IFGRPL AM=VTAM         | VTAM RPL                |
|         | SPACE 1                |                         |
|         | IHASDWA DSECT=YES      | SDWA                    |
|         | END DXC394I            |                         |

## Enhancing the exits to use the MGCRE macro to submit MVS commands

The DXC393I / DXC394I WTO sample exits can be updated in order to automate commands such as:

```
V TCPIP,tcpip_proc,SYSPLEX,DEACTIVATE,DVIPA=vipa'  
V TCPIP,tcpip_proc,SYSPLEX,REACTIVATE,DVIPA=vipa'  
V TCPIP,TELNET,OBEYFILE,DSN=dsname(member).
```

```
***** BUILD AN UTOKEN (USING userid)  
SPACE 1  
LA R4,80 TOKEN LENGTH  
STC R4,WORK_TOKN_DYN SET TOKEN LENGTH  
MVI WORK_TOKN_DYN+1,X'01' SET FORMAT VERSION  
LA R5,RACR_LIST_DYN LOAD ADDRESS OF PLIST  
MVC RACR_LIST_DYN,RACR_LIST COPY PLIST  
RACROUTE REQUEST=TOKENBLD,MF=(E,(R5)),RELEASE=1.9, -  
MSGRTRN=NO,MSGSUPP=NO, -  
SESSION=SYSAS,TRUSTED,REMOTE=NO,USERID=INUSER, -  
TOKNOUT=WORK_TOKN_DYN,WORKA=RACWRK_DYN  
SPACE 1  
***** ISSUE OPERATOR COMMAND  
SPACE 1  
LA R6,L'CMD0TEXT COMMAND LENGTH  
STH R6,CMD_LEN_DYN SET COMMAND LENGTH  
MVC CMD_TEXT_DYN,CMD0TEXT COPY COMMAND  
LA R7,CMD_AREA_DYN ADDRESS OF COMMAND AREA  
SR R8,R8 SET CONSOLE ID TO ZEROES  
LA R9,WORK_TOKN_DYN ADDRESS OF UTOKEN  
MVC MGCRE_LIST_DYN,MGCRE_LIST COPY PLIST  
LA R10,MGCRE_LIST_DYN LOAD ADDRESS OF PLIST  
MGCRE TEXT=(R7),CONSID=(R8),UTOKEN=(R9),MF=(E,(R10))  
  
*  
*  
*  
  
SPACE 1  
***** CONSTANTS USED BY THIS ROUTINE  
SPACE 1  
RACR_LIST RACROUTE REQUEST=TOKENBLD,RELEASE=1.9,WORKA=0,MF=L  
RACR_LIST_L EQU *-RACR_LIST  
SPACE 1  
MGCRE_LIST MGCRE MF=L  
MGCRE_LIST_L EQU *-MGCRE_LIST  
SPACE 1  
INUSER DC AL1(length_of_userid) <<<< USE USERID WITH APPROPRIATE -  
AUTHORITY >>>>  
  
DC C'userid'  
SPACE 1  
CMD0TEXT DC C'V TCPIP,tcpip_proc,SYSPLEX,DEACTIVATE,DVIPA=vipa' -  
<<<< USE APPROPRIATE COMMAND >>>>  
  
*  
*  
*  
  
SPACE 1  
***** FIELDS WHICH MUST BE ADDED TO THE DSECT DATAAREA
```

```
SPACE 1
MGCRE_LIST_DYN DS CL(MGCRE_LIST_L)
CMD_AREA_DYN DS 0F
CMD_LEN_DYN DS H
CMD_TEXT_DYN DS CL49
RACR_LIST_DYN DS CL(RACR_LIST_L)
WORK_TOKN_DYN DS CL80
RACWRK_DYN DS CL512
```

## VIPA types which can be used by ALCS

You could use the DEVICE VIRTUAL and LINK VIRTUAL statements (**static VIPA**) and use an obeyfile to add and remove the VIPA from the stacks as needed so that the VIPA is only active on one stack at a time. Just keep in mind that when that original stack/system recovers the VIPA definition will still be there on that stack (*unless it was only added via obeyfiles*). The same VIPA defined on two different systems will cause IP routing problems.

Or use VARY TCPIP,,SYSPLEX support rather than an obeyfile. What you do is to have a VIPADEFINE definition (**multiple application-instance DVIPA**) in the TCP/IP profile where you want the DVIPA to initially be defined. Then, for another stacks TCP/IP profile, you have a VIPABACKUP definition in its TCP/IP profile for that same DVIPA - this is the place you want to move the DVIPA in case the application needs to get restarted on a different stack. Then, if the application fails on the first TCP/IP stack, before you move it to the second TCP/IP stack, you would issue a

```
V TCPIP,,SYSPLEX,DEACTIVATE,DVIPA=ipaddress
```

on this first stack. That would cause the DVIPA to get deleted from the first stack and automatically created on the second stack (where the VIPABACKUP definition is).

You can update the DXC393I / DXC394I WTO sample exits in order to automate commands like:

```
V TCPIP,tcpip_proc,SYSPLEX,DEACTIVATE,DVIPA=vipa'
```

```
V TCPIP,tcpip_proc,SYSPLEX,REACTIVATE,DVIPA=vipa'
```

```
V TCPIP,xxxxxxx,OBEYFILE,DSN=dsname(member).
```

**End of document**