195

MVS update

December 2002

In this issue

MVS Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: trevore@xephon.com

North American office

Xephon
PO Box 350100
Westminster, CO 80035-0100
USA
Telephone: 303 410 9344

Subscriptions and back-issues

A year's subscription to *MVS Update*, comprising twelve monthly issues, costs £340.00 in the UK; $505.00 in the USA and Canada; £346.00 in Europe; £352.00 in Australasia and Japan; and £350.00 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1999 issue, are available separately to subscribers for £29.00 ($43.50) each including postage.

MVS Update on-line

Code from *MVS Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon.com/mvs; you will need to supply a word from the printed issue.

Editor

Trevor Eddolls

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, EXECs, and other contents of this journal before making any use of it.

Contributions

When Xephon is given copyright, articles published in *MVS Update* are paid for at the rate of £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

Printed in England.

Files allocated by a job, user, or STC

The following program lists in alphabetical order all the files that a job, a TSO user, or a started task has allocated, to what queues, and with what status. It does this by searching the address space chain until it finds the desired name, and then scans the Resource Information Block (RIB) for that address space, listing all the files and queue names. This utility consists of an Assembler program that does the work and a REXX EXEC with the same name to facilitate the module call and pass the argument to it (the argument is the name of the job, user, or STC). To use this utility, assemble the program to a loadlib of your choice, and correct the beginning of the EXEC to have variable loadlib point to it. Put the EXEC in a PDS that belongs to the sysproc concatenation, so you can call it directly by name.

QJOBFILE REXX

```
/* REXX MVS *=========================================================*/
/* QJOBFILE: Query what files a TSO user, a job, or a started task*/
/*           have allocated, and in what queues.                    */
/* Argument: User, job, or STC name.                                 */
/*===================================================================*/
loadlib = 'MY.LOADLIB'      /* loadlib containing qjobfile module */
arg name .
if name = "" then do
   say "QJOBFILE: Enter the name of job, user or started task"
   pull name .
   if name = '' then exit
end
address tso call "'"loadlib"(QJOBFILE)' '"name"'"
exit
```

QJOBFILE ASSEMBLER

```
*====================================================================*
*  QJOBFILE - Query what files are allocated by a job, a TSO user,  *
*             or a started task.                                     *
*  Argument: Fullname of job, user, or STC to search.                *
*  The output is presented with filenames in ascending order.       *
*====================================================================*
&PROGRAM SETC  'QJOBFILE'
```

```
&PROGRAM CSECT
&PROGRAM AMODE 31
&PROGRAM RMODE 24
         STM   R14,R12,12(R13)
         LR    R12,R15
         USING QJOBFILE,R12
         LA    R11,SAVEREGS
         ST    R11,8(R13)
         ST    R13,4(R11)
         LR    R13,R11
         B     GETPARM
         DC    CL16' &PROGRAM V.1.4'
         DC    CL8'&SYSDATE'
*
GETPARM  EQU   *
         L     R7,Ø(R1)            Get parameter address
         LH    R3,Ø(R7)            And length
         LTR   R3,R3
         BZ    EXIT1               Exit if no parm entered
         CH    R3,=H'8'            If parm greater than
         BNH   *+8                 8 bytes, force limit
         LA    R3,8
         SH    R3,=H'1'            Subtract for execute move
         EX    R3,EXMOVPAR         Move parm
         L     R3,CVTPTR           Address CVT
         USING CVT,R3
         L     R2,CVTASVT          Address CVTASVT
         USING ASVT,R2
         L     R8,ASVTMAXU
         LA    R2,ASVTENTY
         L     R6,Ø(R2)            Point to first address space
         USING ASCB,R6             control block
*
ASCBSRCH EQU   *
         L     R1Ø,ASCBJBNI        Jobname address
         LTR   R1Ø,R1Ø             Any address?
         BNZ   CHECKNAM            Yes, jump ahead
         L     R1Ø,ASCBJBNS        Stc or user address
         LTR   R1Ø,R1Ø             Any address?
         BZ    ASCBNEX1            No, go to next
*
CHECKNAM EQU   *
         CLC   Ø(8,R1Ø),JOBNAME    Name equal to parameter?
         BNE   ASCBNEX1            No, get next
         LH    R1Ø,ASCBASID        Yes, store addrspace id
         ST    R1Ø,ADRSPCID
         B     ASCBNEX1
*
ASCBNEXT EQU   *
         TM    Ø(R2),X'8Ø'         Ascb used?
```

```
         BO    ASCBNEX1            No, get next
         L     R6,Ø(R2)            Yes, load pointer
         B     ASCBSRCH            and go search it
*
ASCBNEX1 EQU  *
         LA    R2,4(R2)            Next pointer
         BCT   R8,ASCBNEXT         Loop for asvtmax times
         L     R1Ø,ADRSPCID
         C     R1Ø,=F'-1'          Address space found?
         BNE   GETSTOR             Yes, jump ahead
         TPUT  =C'>>>> Job/Stc/User not found <<<<',32
         B     EXIT1               No, send message and exit
*
GETSTOR  EQU   *
         LA    R8,AREALEN          Get area for output lines
         STORAGE OBTAIN,                                          X
               LENGTH=(R8),                                       X
               ADDR=(R6)
         ST    R6,AREAADDR         R6: output area address
         XR    R8,R8               R8: line counter
*
LOOP1    EQU   *
         L     R3,RISPACEL         Issue gqscan for the address space
         L     R4,ADRSPCID
         GQSCAN AREA=(RISPACE,(R3)),                              X
               SYSNAME=(Ø,(R4)),                                  X
               TOKEN=TOKENØ,                                      X
               SCOPE=ALL
         ST    R15,GQSCANRC        Save return code
         CH    R15,=H'4'           If rc not Ø and not 8, exit
         BE    EXIT1
         CH    R15,=H'8'
         BH    EXIT1
         LR    R11,R1              R11 number of ribs
         LR    R5,RØ               R5 fixed rib length
         SRL   R5,16
         LR    R2,RØ
         SLL   R2,16
         SRL   R2,16
         L     R7,RIBSPADR         Address ribspace
         USING RIB,R7
*
LOOP2    EQU   *                   Rib loop
         LR    R3,R7
         AR    R3,R5               Add fixed rib length
         AH    R3,RIBVLEN          Add variable length
         USING RIBE,R3             Address first rib
         L     R9,RIBNRIBE         R9 number of ribes in rib
         LR    R4,R7
         AR    R4,R5               Address rib variable portion
```

```
         USING RIBVAR,R4
         MVC   OUTLIN(8Ø),=CL8Ø' ' Clear output line
         XR    R1Ø,R1Ø
         IC    R1Ø,RIBRNMLN        Get rname (filename) length
         CH    R1Ø,=H'44'
         BL    *+8
         LA    R1Ø,44
         SH    R1Ø,=H'1'           And prepare for move execute
         EX    R1Ø,EXMOVNAM        Move filename to output line
*
LOOP3    EQU   *                   Rib extent loop
         MVC   OUTLIN2,RIBQNAME    Ribe queue name
         TM    RIBERFLG,RIBETYPE   Test bit type: 1=shr  Ø=old
         BO    TYPESHR
         MVC   OUTLIN3,=CL3'OLD'
         B     STATUS
*
TYPESHR  MVC   OUTLIN3,=CL3'SHR'
*
STATUS   TM    RIBESFLG,RIBESTAT   Test bit status: 1=use  Ø=wait
         BO    STATUSIN
         MVC   OUTLIN4,=CL7'Waiting'
         B     MOVELINE
*
STATUSIN MVC   OUTLIN4,=CL7'Using'
*
MOVELINE EQU   *
         MVC   Ø(8Ø,R6),OUTLIN     Move output line to outarea
         LA    R6,8Ø(Ø,R6)         Inc outarea pointer
         LA    R8,1(Ø,R8)          Inc line counter
         AR    R3,R2               Point to next ribe
         BCT   R9,LOOP3            and loop to next
         LR    R7,R3               Point next rib
         BCT   R11,LOOP2           and loop to next
         CLC   GQSCANRC,=F'8'      Overflow? (gqscan retcode was 8)
         BE    LOOP1               Yes, gqscan once more.
*
LINESORT EQU   *
         ST    R8,NUMLINES         Store number of output lines
         L     R6,AREAADDR         Sort output lines by filename
*                                  R6 and R7 point to table
SORTØ    EQU   *                   R8 and R9 count the loops
         LA    R7,8Ø(Ø,R6)
         LR    R9,R8
         SH    R9,=H'1'
         LTR   R9,R9
         BZ    SENDHEAD
*
SORT1    EQU   *
         CLC   Ø(44,R6),Ø(R7)      Compare filenames
```

```
         BNH   SORT2
         MVC   OUTLIN,Ø(R7)        If greater, switch lines
         MVC   Ø(8Ø,R7),Ø(R6)
         MVC   Ø(8Ø,R6),OUTLIN
*
SORT2    EQU   *
         LA    R7,8Ø(Ø,R7)         Increment inner pointer R7
         BCT   R9,SORT1            Inner loop counter R9
         LA    R6,8Ø(Ø,R6)         Increment outer pointer R6
         BCT   R8,SORTØ            Outer loop counter R8
*
SENDHEAD EQU   *
         STFSMODE ON,INITIAL=YES   Set full screen on
         STTMPMD  ON
         STLINENO LINE=1           Clear screen
         TPUT  HEADLIN1,79         Write header line
         TPUT  =C' ',1             Write space line
         L     R6,AREAADDR         Retrieve output area address
         L     R8,NUMLINES         And number of lines
*
SENDLINE EQU   *
         TPUT  (R6),79             Write line
         LA    R6,8Ø(Ø,R6)         Point to next
         BCT   R8,SENDLINE         Loop for existing lines
*
EXIT     EQU   *
         STFSMODE OFF              Fullscreen mode off
         L     R6,AREAADDR
         L     R8,NUMLINES
         STORAGE RELEASE,          Release storage                    X
               LENGTH=(R8),                                         X
               ADDR=(R6)
*
EXIT1    EQU   *
         L     R15,GQSCANRC        Get gqscan return code
         L     R13,4(R13)          Restore other registers
         L     R14,12(R13)
         LM    RØ,R12,2Ø(R13)
         BR    R14
*=================================================================*
*  Execute instructions, workareas, and mapping macros            *
*=================================================================*
EXMOVPAR MVC   JOBNAME(Ø),2(R7)
EXMOVNAM MVC   OUTLIN1(Ø),RIBRNAME
*
         LTORG
OUTLIN   DS    ØCL8Ø
         DS    CL1
OUTLIN1  DS    CL44
         DS    CL2
OUTLIN2  DS    CL8
```

```
         DS    CL2
OUTLIN3  DS    CL3
         DS    CL3
OUTLIN4  DS    CL7
         DS    CL2Ø
HEADLIN1 DC    CL44' Filename'
         DC    CL4Ø'  Queuename  Disp  Status'
JOBNAME  DC    CL8' '
*
         DS    ØF
SAVEREGS DS    18F
AREALEN  DC    F'64ØØØ'
AREAADDR DS    F
NUMLINES DS    F
ADRSPCID DC    F'-1'
GQSCANRC DS    F
TOKENØ   DC    F'Ø'
RIBSPADR DC    XL1'8Ø'
         DC    AL3(RISPACE)
RISPACEL DC    F'5ØØØ'
RISPACE  DS    5ØØØC
*
         CVT   DSECT=YES
         ISGRIB
         IHAASCB
         IHAASVT
         YREGS
         END
```

Systems Programmer
(Portugal)

Keeping track of 'lost' ASIDs

INTRODUCTION

Address spaces that terminate with active cross memory binds are marked non-reusable. This is indicated by the following message:

```
IEF352I ADDRESS SPACE UNAVAILABLE
```

The ASVT entry remains non-reusable until all of the address spaces that the job had cross memory binds with have also

ended. If the cross memory binds are with an address space such as CONSOLE, the ASID is 'lost' until the next IPL. This is common when stopping ADABAS, DB2, and MQM.

The RSVNONR parameter specified in IEASYS*xx* specifies the number of entries in the ASVT that are to be reserved for replacing entries marked as non-reusable. There are no MVS, OS/390, or z/OS commands that allow you to display how many ASIDs are lost, or how many entries in RSVNONR have been used.

This program will write out the names of all ASIDs in the system, their ASCB address, their ASID number in hex and in decimal, the number of non-reusable ASIDs, and IEASYS*xx* specified (and used) values for MAXUSER, RSVSTRT, and RSVNONR.

ASIDLIST PROGRAM SOURCE

```
         TITLE 'ASIDLIST - LIST ASIDS IN THE SYSTEM'
         PRINT NOGEN
*********************************************************************
* THIS PROGRAM WILL WRITE OUT THE NAMES OF ALL ADDRESS SPACE        *
* IDS IN THE SYSTEM, THEIR ASCB ADDRESS, THEIR ASID NUMBER IN       *
* HEX AND IN DECIMAL AS SHOWN IN THE SAMPLE BELOW:                  *
*                                                                   *
*   ASCB FOUND AT 00FD4100: *MASTER* - ASID X'0001' (   1 IN DECIMAL) *
*   ASCB FOUND AT 00F4BE80: PCAUTH   - ASID X'0002' (   2 IN DECIMAL) *
*   ASCB FOUND AT 00F4D700: RASP     - ASID X'0003' (   3 IN DECIMAL) *
*   ASCB FOUND AT *NONREUS: *NONREUS - ASID X'000E' (  14 IN DECIMAL) *
*   ASCB FOUND AT *AVALABL: *AVALABL - ASID X'0037' (  55 IN DECIMAL) *
*   ASCB FOUND AT *AVALABL: *AVALABL - ASID X'NNNN' (NNNN IN DECIMAL) *
*                                                                   *
* THIS PROGRAM ALSO DISPLAYS TOTALS AS SHOWN BELOW:                 *
*                                                                   *
*       TOTAL ADDRESS SPACES IN THE SYSTEM:               NNNN      *
*       TOTAL ACTIVE ADDRESS SPACES IN THE SYSTEM:        NNNN      *
*       TOTAL AVAILABLE ADDRESS SPACES IN THE SYSTEM:     NNNN      *
*       TOTAL NON-REUSABLE ADDRESS SPACES IN THE SYSTEM:  NNNN      *
*                                                                   *
*               ASID USAGE FROM ASVT                                *
*                                                                   *
*       MAXUSER FROM IEASYSXX:  NNNN                                *
*                IN USE ASIDS:  NNNN                                *
*              AVAILABLE ASIDS:  NNNN                               *
*                                                                   *
*       RSVSTRT FROM IEASYSXX:  NNNN                                *
```

```
*              RSVSTRT IN USE:  NNNN                                  *
*           RSVSTRT AVAILABLE:  NNNN                                  *
*                                                                     *
*       RSVNONR FROM IEASYSXX:  NNNN                                  *
*              RSVNONR IN USE:  NNNN                                  *
*           RSVNONR AVAILABLE:  NNNN                                  *
*                                                                     *
*       NON-REUSABLE ASIDS   :  NNNN                                  *
*                                                                     *
* THE DEFAULT OUTPUT IS 55 LINES PER PAGE. AN OPTIONAL 2 DIGIT INPUT  *
* PARAMETER MAY BE SPECIFIED TO CHANGE THE DEFAULT LINES PER PAGE.    *
***********************************************************************
*** SAMPLE JCL:
***
*** //ASIDLIST JOB (ACCT),'COUNT JOBS',CLASS=S
*** //STEP1 EXEC PGM=ASIDLIST,PARM=58
*** //SYSPRINT  DD  SYSOUT=*
*** //
***
*
*  REGISTER EQUATES AND USAGE
*
RØØ      EQU   Ø                  LINKAGE REGISTER
RØ1      EQU   1                  INITIAL POINTER TO INPUT PARM
RØ2      EQU   2                  WORK REG
RØ3      EQU   3                  POINTS TO PARM / MAX # ASVT ENTRIES
RØ4      EQU   4                  WORK - POINTS TO CURRENT ADDR IN ASVT
RØ5      EQU   5                  WORK REG
RØ6      EQU   6                  WORK REG - USED FOR BAL TO PRINT RTNS
RØ7      EQU   7                  POINTS TO START OF ASVTENTY
RØ8      EQU   8                  ASID COUNTER
RØ9      EQU   9                  BASE REG FOR ASVT
R1Ø      EQU   1Ø                 BASE REG FOR ASCB
R11      EQU   11                 2ND BASE REG
R12      EQU   12                 BASE REGISTER
R14      EQU   14                 LINKAGE REGISTER (RETURN ADDRESS)
R15      EQU   15                 LINKAGE REGISTER (ENTRY POINT)
*
ASIDLIST CSECT
         B     START-ASIDLIST(R15)
         DC    AL1(START-*)
         DC    C'ASIDLIST &SYSDATE &SYSTIME '
         DC    C'*** AUTHOR: MARK ZELDEN ***'
START    BAKR  R14,RØØ            BRANCH AND STACK (LINKAGE STACK)
         LR    R12,R15            SET UP ADDRESSABILITY
         LA    R11,2Ø48(R12)      SET UP ADDRESSABILITY TO 2ND
         LA    R11,2Ø48(R11)        BASE REGISTER
         USING ASIDLIST,R12,R11   SET UP BASE REGISTERS
* ====================================================================
**********************************************************************
```

```
* PROCESS INPUT PARM (IF THERE IS ONE)
*****************************************************************
         L     RØ3,Ø(RØ1)          POINT TO INPUT PARM
         CLC   Ø(2,RØ3),=H'Ø'      IS LENGTH=Ø (NO PARM) ?
         BNE   USEPARM             NO, BRANCH AND USE PARM VALUE
         MVC   MAXLINES,=PL2'55'   YES, MOVE DEFAULT MAX LINES PER PAGE
         B     OPEN                BRANCH
USEPARM  CLC   Ø(2,RØ3),=H'2'      IS LENGTH=2  ?
         BNE   BADPARM             NO, BRANCH
         MVC   PARMLINE,2(RØ3)     MOVE PARM VALUE
         PACK  MAXLINES,PARMLINE   PACK IT
*****************************************************************
* OPEN SYSPRINT AND INITIALIZE COUNTERS
*****************************************************************
OPEN     OPEN  (SYSPRINT,(OUTPUT)) OPEN SYSPRINT FILE
         ZAP   TOTASIDS,=P'Ø'      ZERO OUT TOTAL ASID COUNTER
         ZAP   TOTACTIV,=P'Ø'      ZERO OUT TOTAL ACTIVE ASID COUNTER
         ZAP   TOTAVAL,=P'Ø'       ZERO OUT TOTAL AVAILABLE ASID COUNTER
         ZAP   TOTNONR,=P'Ø'       ZERO OUT TOTAL NON-REUSABLE COUNTER
         ZAP   LINCOUNT,=P'Ø'      ZERO CURRENT LINE # ON PAGE COUNTER
         ZAP   PAGCOUNT,=P'1'      INITIALIZE PAGE COUNTER
         LA    RØ8,1               INITIALIZE ASID COUNTER TO 1
         BAL   RØ6,PUTTITLE        BRANCH TO PRINT TITLE SUB-ROUTINE
*****************************************************************
* POINT TO ASVT
*****************************************************************
         L     RØ9,CVTPTR          POINT TO CVT  - X'1Ø'
         USING CVT,RØ9             MAP CVT
         L     RØ9,CVTASVT         POINT TO ASVT
         DROP  RØ9                 TELL ASMBLR TO STOP USING RØ9 FOR CVT
         USING ASVT,RØ9            MAP ASVT
         LA    RØ4,ASVTENTY        POINT TO FIRST ENTRY IN TABLE
         LR    RØ7,RØ4             SAVE ADDRESS IN R7 FOR LATER
         L     RØ3,ASVTMAXU        LOAD MAX NUMBER OF ENTRIES
ASVTLOOP DS    ØH
*****************************************************************
* IS A NEW TOP OF FORM IS NEEDED ?
*****************************************************************
TITLECHK DS    ØH
         CP    LINCOUNT,MAXLINES   DO WE NEED A NEW PAGE?
         BL    SAMEPAGE            NO, BRANCH
         BAL   RØ6,PUTTITLE        BRANCH TO PRINT TITLE SUB-ROUTINE
SAMEPAGE DS    ØH
*****************************************************************
* THIS ROUTINE CHECKS EACH ASVT ENTRY.
* IF THE HIGH ORDER BIT IS ON, THE ENTRY IS THE ADDRESS OF THE
* NEXT AVAILABLE ASID (OR THE LAST ENTRY IF ZEROS).
* IF THE HIGH ORDER BIT IS NOT ON, THE ENTRY IS THE ADDRESS
* OF THE ASCB FOR THAT ENTRY. IF THE HIGH ORDER BIT IS ON AND
* THE ENTRY CONTAINS THE ADDRESS OF MASTER'S ASCB, THEN THE ASID
```

```
* IS NON-REUSABLE.
*********************************************************************
         TM    Ø(RØ4),ASVTAVAL    IS THIS AN ASSIGNED  ASCB
         BNO   CHKASCB            YES, BRANCH
         L     RØ5,Ø(RØ4)         SAVE ADDRESS
         SL    RØ5,=X'8ØØØØØØØ'    ZERO OUT HIGH ORDER BIT
         CR    RØ5,RØ7            IS THIS A NON-REUSABLE ASID?
         BNE   AVALAS             MUST BE AVAILABLE, BRANCH
         MVC   JOBNAME,=C'*NONREUS' MOVE 'NONREUS' INTO JOBNAME
         AP    TOTNONR,=P'1'      ADD 1 TO TOTAL NON-REUSABLE COUNTER
         BAL   RØ6,PUTPRTLN       BRANCH TO PRINT SUB-ROUTINE
         LA    RØ4,4(,RØ4)        NO, POINT TO NEXT ENTRY IN ASVT
         BCT   RØ3,ASVTLOOP       GO CHECK NEXT ASVT ENTRY
         B     TOTALS             NO MORE ENTRIES - BRANCH
AVALAS   DS    ØH
         MVC   JOBNAME,=C'*AVALABL' MOVE 'AVAILABLE' INTO JOBNAME
         AP    TOTAVAL,=P'1'      ADD 1 TO TOTAL AVAILABLE COUNTER
         BAL   RØ6,PUTPRTLN       BRANCH TO PRINT SUB-ROUTINE
         LA    RØ4,4(,RØ4)        NO, POINT TO NEXT ENTRY IN ASVT
         BCT   RØ3,ASVTLOOP       GO CHECK NEXT ASVT ENTRY
         B     TOTALS             NO MORE ENTRIES - BRANCH
*********************************************************************
* CHECK ASCB FOR JOB OR START/LOGON/MOUNT
*********************************************************************
CHKASCB  L     R1Ø,Ø(RØ4)         POINT TO ASCB
         USING ASCB,R1Ø           MAP IT
         L     RØ5,ASCBJBNI       POINT TO JOBNAME
         C     RØ5,=F'Ø'          WAS THIS A START/MOUNT/LOGON ?
         BE    NOTAJOB            YES, BRANCH
         MVC   JOBNAME,Ø(RØ5)     MOVE JOBNAME INTO MSG
         AP    TOTACTIV,=P'1'     ADD 1 TO TOTAL ACTIVE COUNTER
         BAL   RØ6,PUTPRTLN       BRANCH TO PRINT SUB-ROUTINE
         LA    RØ4,4(,RØ4)        POINT TO NEXT ENTRY IN ASVT
         BCT   RØ3,ASVTLOOP       GO CHECK NEXT ASVT ENTRY
         B     TOTALS             NO MORE ENTRIES - BRANCH
NOTAJOB  DS    ØH
         L     RØ5,ASCBJBNS       POINT TO START/MOUNT/LOGON NAME
         C     RØ5,=F'Ø'          NAME PRESENT ?
         BNE   MOVESNAM           YES, BRANCH
         MVC   JOBNAME,=C'*STRTING' MOVE 'STRTING' INTO JOBNAME
         B     SKIPSNAM
MOVESNAM MVC   JOBNAME,Ø(RØ5)     MOVE JOBNAME INTO MSG
SKIPSNAM AP    TOTACTIV,=P'1'     ADD 1 TO TOTAL ACTIVE COUNTER
         BAL   RØ6,PUTPRTLN       BRANCH TO PRINT SUB-ROUTINE
         LA    RØ4,4(,RØ4)        POINT TO NEXT ENTRY IN ASVT
         BCT   RØ3,ASVTLOOP       GO CHECK NEXT ASVT ENTRY
         B     TOTALS             NO MORE ENTRIES - BRANCH
*********************************************************************
* SUB ROUTINE TO PRINT TITLE LINES ON TOP OF PAGE
*********************************************************************
```

```
PUTTITLE MVC   EDPAGNUM,EDMASK    MOVE EDIT WORD TO OUTPUT
         ED    EDPAGNUM,PAGCOUNT  MAKE PAGE NUMBER COUNT PRINTABLE
         MVC   PAGENUM(2),EDPAGNUM+2
         PUT   SYSPRINT,TITLELN1  WRITE
         PUT   SYSPRINT,TITLELN2    TITLE
         PUT   SYSPRINT,BLANKLIN  WRITE BLANK LINE
         AP    PAGCOUNT,=P'1'     ADD 1 TO PAGE NUMBER COUNTER
         ZAP   LINCOUNT,=P'2'     INITIALIZE LINE NUMBER COUNTER TO 2
         BR    RØ6                RETURN
*****************************************************************
* SUB ROUTINE TO WRITE A PRINT LINE
*****************************************************************
PUTPRTLN LR    RØ5,RØ8            LOAD ASID NUMBER
         CVD   RØ5,CVDWORK        CONVERT TO DECIMAL
         MVC   EDASID,EDMASK2     MOVE EDIT MASK
         ED    EDASID,CVDWORK+5   UNPACK AND EDIT
         MVI   ASIDHEX+6,C'('     MOVE "(" TO PRINTOUT
         STCM  RØ8,B'ØØ11',WORK3  STORE "HEX" ASID
         UNPK  WORK5,WORK3        ADD ZONES
         TR    WORK5(4),HEXTAB-C'Ø'   TRANSLATE TO CHARACTERS
         MVC   ASIDHEX(4),WORK5   MOVE HEX ASID
*================================================
         CLC   JOBNAME,=C'*NONREUS' IS IT A NON-REUSABLE ASID?
         BNE   CHKAVAL
         MVC   ASCBADDR,=C'*NONREUS'
         B     SKIPCVRT
CHKAVAL  CLC   JOBNAME,=C'*AVALABL' IS IT AN AVAILABLE ASID
         BNE   CVRTADDR
         MVC   ASCBADDR,=C'*AVALABL'
         B     SKIPCVRT
CVRTADDR ST    R1Ø,WORK4          STORE ASCB ADDR X'ABCDEFAB'
         MVC   WORK5(4),WORK4     ABCDEFABØØ
         UNPK  WORK9,WORK5        FAFBFCFDFEFFFAFBØØ
         TR    WORK9(8),HEXTAB-C'Ø'   C1C2C3C4C5C6C1C2ØØ
         MVC   ASCBADDR(8),WORK9  C'ABCDEFAB'
SKIPCVRT PUT   SYSPRINT,ACTMSG    WRITE JOB ACTIVE MSG
         MVC   JOBNAME,=CL8' '    CLEAR JOBNAME
         AP    TOTASIDS,=P'1'     ADD 1 TO TOTAL COUNTER
         AP    LINCOUNT,=P'1'     ADD 1 TO LINE NUMBER COUNTER
         AH    RØ8,=X'ØØØ1'       ADD 1 TO ASID NUMBER
         BR    RØ6                RETURN
*****************************************************************
* PUT OUT TOTAL MESSAGES
*****************************************************************
TOTALS   DS    ØH
         L     RØ5,ASVTMAXI       MAXUSERS FROM ASVT
         CVD   RØ5,CVDWORK        CONVERT TO DECIMAL
         MVC   EDTOTMX,EDMASK2    MOVE EDIT MASK
         ED    EDTOTMX,CVDWORK+5  UNPACK AND EDIT
         L     RØ6,ASVTAAV        AVAILABLE FROM ASVT
```

```
         CVD   RØ6,CVDWORK        CONVERT TO DECIMAL
         MVC   EDTOTAVA,EDMASK2    MOVE EDIT MASK
         ED    EDTOTAVA,CVDWORK+5  UNPACK AND EDIT
         SLR   RØ5,RØ6             CALCULATED IN USE ASIDS
         CVD   RØ5,CVDWORK         CONVERT TO DECIMAL
         MVC   EDTOTAVI,EDMASK2    MOVE EDIT MASK
         ED    EDTOTAVI,CVDWORK+5  UNPACK AND EDIT
         L     RØ5,ASVTSTRT        START/SASI FROM ASVT (RSVSTRT)
         CVD   RØ5,CVDWORK         CONVERT TO DECIMAL
         MVC   EDTOTST,EDMASK2     MOVE EDIT MASK
         ED    EDTOTST,CVDWORK+5   UNPACK AND EDIT
         L     RØ6,ASVTAST         AVAILABLE START/SASI FROM ASVT
         CVD   RØ6,CVDWORK         CONVERT TO DECIMAL
         MVC   EDTOTSTA,EDMASK2    MOVE EDIT MASK
         ED    EDTOTSTA,CVDWORK+5  UNPACK AND EDIT
         SLR   RØ5,RØ6             CALCULATED IN USE SASI
         CVD   RØ5,CVDWORK         CONVERT TO DECIMAL
         MVC   EDTOTSTI,EDMASK2    MOVE EDIT MASK
         ED    EDTOTSTI,CVDWORK+5  UNPACK AND EDIT
         L     RØ5,ASVTNONR        NON-RESUSABLE FROM ASVT (RSVNONR)
         CVD   RØ5,CVDWORK         CONVERT TO DECIMAL
         MVC   EDTOTNR,EDMASK2     MOVE EDIT MASK
         ED    EDTOTNR,CVDWORK+5   UNPACK AND EDIT
         L     RØ6,ASVTANR         AVAILABLE NON-REUSABLE FROM ASVT
         CVD   RØ6,CVDWORK         CONVERT TO DECIMAL
         MVC   EDTOTNRA,EDMASK2    MOVE EDIT MASK
         ED    EDTOTNRA,CVDWORK+5  UNPACK AND EDIT
         SLR   RØ5,RØ6             CALCULATED IN USE NON-REUSABLE
         CVD   RØ5,CVDWORK         CONVERT TO DECIMAL
         MVC   EDTOTNRI,EDMASK2    MOVE EDIT MASK
         ED    EDTOTNRI,CVDWORK+5  UNPACK AND EDIT
*
         BAL   RØ6,PUTTITLE        BRANCH TO PRINT TITLE SUB-ROUTINE
         MVC   EDTOTAS,EDMASK2     MOVE EDIT WORD TO OUTPUT
         ED    EDTOTAS,TOTASIDS    MAKE TOTAL COUNT PRINTABLE
         MVC   EDTOTACT,EDMASK2    MOVE EDIT WORD TO OUTPUT
         ED    EDTOTACT,TOTACTIV   MAKE TOTAL COUNT PRINTABLE
         MVC   EDTOTAV,EDMASK2     MOVE EDIT WORD TO OUTPUT
         ED    EDTOTAV,TOTAVAL     MAKE TOTAL COUNT PRINTABLE
         MVC   EDTOTNOR,EDMASK2    MOVE EDIT WORD TO OUTPUT
         ED    EDTOTNOR,TOTNONR    MAKE TOTAL COUNT PRINTABLE
*
         MVC   TOT2NOR,EDTOTNOR    MOVE EDITED NUMBER OR NON-REUSE
*
         PUT   SYSPRINT,TOTALLN1   WRITE TOTAL1 HEADER LINE
         PUT   SYSPRINT,BLANKLIN   WRITE BLANK LINE
         PUT   SYSPRINT,TOTASMG    WRITE TOTAL ASID MSG
         PUT   SYSPRINT,TOTACTMG   WRITE TOTAL ACTIVE ASID MSG
         PUT   SYSPRINT,TOTAVAMG   WRITE TOTAL AVAILABLE ASID MSG
         PUT   SYSPRINT,TOTNORMG   WRITE TOTAL NON-REUSABLE ASID MSG
```

```
         PUT   SYSPRINT,BLANKLIN  WRITE BLANK LINE
         PUT   SYSPRINT,BLANKLIN  WRITE BLANK LINE
         PUT   SYSPRINT,TOTALLN2  WRITE TOTAL2 HEADER LINE
         PUT   SYSPRINT,BLANKLIN  WRITE BLANK LINE
         PUT   SYSPRINT,TOTMAXU   WRITE ASVT MAXUSER LINE
         PUT   SYSPRINT,TOTAVI    WRITE ASVT TOTAL IN USE LINE
         PUT   SYSPRINT,TOTAVA    WRITE ASVT TOTAL AVAILABLE LINE
         PUT   SYSPRINT,BLANKLIN  WRITE BLANK LINE
         PUT   SYSPRINT,TOTSASI   WRITE ASVT RSVSTRT LINE
         PUT   SYSPRINT,TOTSASII  WRITE ASVT RSVSTRT IN USE LINE
         PUT   SYSPRINT,TOTSASIA  WRITE ASVT RSVSTRT AVAILABLE LINE
         PUT   SYSPRINT,BLANKLIN  WRITE BLANK LINE
         PUT   SYSPRINT,TOTNR     WRITE ASVT RSVNONR LINE
         PUT   SYSPRINT,TOTNRI    WRITE ASVT RSVNONR IN USE LINE
         PUT   SYSPRINT,TOTNRA    WRITE ASVT RSVNONR AVAILABLE LINE
         PUT   SYSPRINT,BLANKLIN  WRITE BLANK LINE
         PUT   SYSPRINT,TOTNONRU  WRITE ASVT NON-REUSABLE ASID LINE
         B     RETURN             GO END
BADPARM  WTO   'ASIDLIST - NUMBER OF LINES PER PAGE IN PARM MUST BE 2 DX
               IGITS - JOB CANCELLED',ROUTCDE=11
         ABEND Ø1,REASON=Ø        UØØØ1 ABEND  - NO DUMP
RETURN   CLOSE (SYSPRINT)         CLOSE FILES
         LA   R15,Ø               SET RETURN CODE TO ZERO
         PR                       PROGRAM RETURN (LINKAGE STACK)
         EJECT
* ================================================================
SYSPRINT DCB   DDNAME=SYSPRINT,DSORG=PS,MACRF=PM,RECFM=FBA,           X
               LRECL=133,BLKSIZE=399Ø
PARMLINE DS    ZL2                PARM VALUE
MAXLINES DS    PL2                MAXIMUM LINES PER PAGE
CVDWORK  DS    D                  WORK AREA FOR CVD
WORK3    DS    CL3                WORK AREA FOR HEX TO CHAR
WORK4    DS    F                  WORK AREA FOR HEX TO CHAR
WORK5    DS    CL5                WORK AREA FOR HEX TO CHAR
WORK9    DS    CL9                WORK AREA FOR HEX TO CHAR
HEXTAB   DC    C'Ø123456789ABCDEF' TRANSLATION TABLE FOR HEX TO CHAR
TOTASIDS DS    PL3                TOTAL # OF ASIDS
TOTACTIV DS    PL3                TOTAL # OF ACTIVE ASIDS
TOTAVAL  DS    PL3                TOTAL # OF AVAILABLE ASIDS
TOTNONR  DS    PL3                TOTAL # OF NON-REUSABLE ASIDS
LINCOUNT DS    PL2                CURRENT LINE COUNT ON PAGE
PAGCOUNT DS    PL2                CURRENT PAGE NUMBER COUNT
EDPAGNUM DS    CL4                EDITED PAGE NUMBER
BLANKLIN DC    CL133' '
TITLELN1 DC    CL5Ø'1      A D D R E S S   S P A C E   I D   L I S '
         DC    CL2Ø'T            PAGE - '
PAGENUM  DS    CL2                PAGE NUMBER
         DC    CL61' '            FILLER
TITLELN2 DC    CL5Ø'+     ________________________________________________'
         DC    CL5Ø'_                                                     '
```

```
         DC    CL33' '            FILLER
TOTALLN1 DC    CL50'                 PROGRAM TOTALS                  '
         DC    CL50'                                                 '
         DC    CL33' '            FILLER
ACTMSG   DC    CL15' ASCB FOUND AT '
ASCBADDR DS    CL8                EDITED ASCB ADDRESS
         DC    CL2': '
JOBNAME  DC    CL8' '
         DC    CL10' - ASID X'''
ASIDHEX  DS    CL4                ASID IN HEX CHAR
         DC    CL1''''
EDASID   DS    CL6                EDITED ASID IN DECIMAL
         DC    CL12' IN DECIMAL)'
         DC    CL67'        '     FILLER
EDMASK   DC    X'40202120'
EDMASK2  DC    X'402020202120'
TOTASMG  DC    CL49' TOTAL ADDRESS SPACES IN THE SYSTEM:             '
EDTOTAS  DS    CL6
         DC    CL78' '            FILLER
TOTACTMG DC    CL49' TOTAL ACTIVE ADDRESS SPACES IN THE SYSTEM:      '
EDTOTACT DS    CL6
         DC    CL78' '            FILLER
TOTAVAMG DC    CL49' TOTAL AVAILABLE ADDRESS SPACES IN THE SYSTEM:   '
EDTOTAV  DS    CL6
         DC    CL78' '            FILLER
TOTNORMG DC    CL49' TOTAL NON-REUSABLE ADDRESS SPACES IN THE SYSTEM:'
EDTOTNOR DS    CL6
         DC    CL78' '            FILLER
TOTALLN2 DC    CL49'              ASID USAGE FROM ASVT               '
         DC    CL49'                                                 '
         DC    CL35' '            FILLER
TOTMAXU  DC    CL23' MAXUSER FROM IEASYSXX:'
EDTOTMX  DS    CL6
         DC    CL104' '            FILLER
TOTAVI   DC    CL23'          IN USE ASIDS:'
EDTOTAVI DS    CL6
         DC    CL104' '            FILLER
TOTAVA   DC    CL23'       AVAILABLE ASIDS:'
EDTOTAVA DS    CL6
         DC    CL104' '            FILLER
TOTSASI  DC    CL23' RSVSTRT FROM IEASYSXX:'
EDTOTST  DS    CL6
         DC    CL104' '            FILLER
TOTSASII DC    CL23'        RSVSTRT IN USE:'
EDTOTSTI DS    CL6
         DC    CL104' '            FILLER
TOTSASIA DC    CL23'     RSVSTRT AVAILABLE:'
EDTOTSTA DS    CL6
         DC    CL104' '            FILLER
TOTNR    DC    CL23' RSVNONR FROM IEASYSXX:'
```

```
EDTOTNR  DS    CL6
         DC    CL1Ø4' '            FILLER
TOTNRI   DC    CL23'        RSVNONR IN USE:'
EDTOTNRI DS    CL6
         DC    CL1Ø4' '            FILLER
TOTNRA   DC    CL23'     RSVNONR AVAILABLE:'
EDTOTNRA DS    CL6
         DC    CL1Ø4' '            FILLER
TOTNONRU DC    CL23' NON-REUSABLE ASIDS   :'
TOT2NOR  DS    CL6
         DC    CL1Ø4' '            FILLER
         LTORG
         CVT DSECT=YES
         IHAASVT
         IHAASCB
         END
```

ASIDLIST EXECUTION JCL

```
//ASIDLIST JOB (ACCT),'LOST ASIDS',CLASS=M
//STEP1 EXEC PGM=ASIDLIST,PARM=58
//SYSPRINT  DD  SYSOUT=*
//
```

Mark Zelden
Systems Programmer (USA)

SUBMIT option for part of JCL using labels

INTRODUCTION

I've recently been extending the funtionality of a product. This product is basically a REXX procedure which, in a loop, calls several ISPF panels and then, using the Job Tailoring services of ISPF, generates JCL. This JCL has many steps, some of which fail to complete successfully after submission. Often, I need to re-submit only a specific part of this JCL to test it further. Up until recently, not wanting to damage my original JCL, I've been going into the spooled output in SDSF using option SJ, editing out unwanted JCL steps, and re-submitting the end

product to test the specific parts further. I've now found out that there is a simpler method using labels in ISPF Edit/View.

METHOD

The method is to quite simply mark the start and end positions with labels in the number columns in ISPF Edit and then give the command **SUBMIT label1 label2** on the command line. This will generate a JOB with the default JOBCARD and the JCL between the two labels.

SYNTAX

The command syntax is:

```
SUBMIT [range] [X]
               [NX]
```

where:

- range – two labels that define the first and last lines to be submitted. The defaults are the editor-defined .ZFIRST and .ZLAST labels.
- X – submits only those lines that are excluded from the display.
- NX – submits only those lines that are not excluded from the display.

For example:

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

ISREDDE2   AL13745.JCL.CNTL(BEIS) - 01.01            Columns 00001 00072
Command ===> SUBMIT A. B.                               Scroll ===> CSR
****** ************************ Top of Data ****************************
000001 //IKJEFT01 EXEC PGM=IKJEFT01,PARM='CONVERT EBCDIC'
000002 //SYSTSPRT DD SYSOUT=*
000003 //SYSTSIN  DD DUMMY
000004 //SYSPRINT DD SYSOUT=*
000005 //SYSOUT   DD SYSOUT=*
000006 //SYSPROC  DD DSN=HBRLSB0.TCPIP.JCL,DISP=SHR
```

```
000007 //*
000008 //EBCDIC   DD DSN=HBRLSB0.STROBE.R231.H01AC054.X002D001,DISP=SHR
000009 //*
000010 //ASCII    DD DSN=HBRLSB0.STROBE.R231.H01AC054.Y002D001,
000011 //         DISP=(,CATLG,DELETE),
000012 //         SPACE=(TRK,(10,10),RLSE),RECFM=VB,LRECL=1028,BLKSIZE=0
000013 //*
000014 //INDICE   DD DSN=HBRLSB0.ISTROBE.INDICE,DISP=(,CATLG,DELETE),
000015 //         SPACE=(TRK,(10,10),RLSE),RECFM=VB,LRECL=1028,BLKSIZE=0
000016 //*
000017 //INDICENV DD DSN=HBRLSB0.ISTROBE.INDICENV,DISP=(,CATLG,DELETE),
000018 //         SPACE=(TRK,(10,10),RLSE),RECFM=VB,LRECL=1028,BLKSIZE=0
000019 //*
000020 //INDICEBK DD DSN=HBRLSB0.ISTROBE.INDICEBK,DISP=SHR
000021 //*
000022 //*
A.0023 //FTPSTEP EXEC PGM=FTP,COND=(8,LE),PARM='172.22.1.200 /(EXIT'  $|
000024 //SYSFTPD  DD *                                                 |
000025 CLIENTERRCODES TRUE                                            |
000026 //SYSPRINT DD SYSOUT=*                                        |
000027 //OUTPUT   DD SYSOUT=*                                        |
000028 //INPUT    DD *                                               |
000029 user                                                          |
000030 password                                                      |
000031 CD /u/p9/hbrlsb0/profiles/                                    |
000032 BINARY                                                        |
000033 SENDSITE                                                      |
000034 PUT'HBRLSB0.ISTROBE.INDICE' profile_list.xml                  |
000035 MK TCPIP                                                      |
000036 CD TCPIP                                                      |
000037 PUT'HBRLSB0.STROBE.R231.H01AC054.Y002D001' profile.xml        |
000038 CLOSE                                                         |
000039 QUIT                                                          |
B.0040 /*                                                          $|
000045 //ICEGENER EXEC PGM=ICEGENER
000046 //SYSPRINT DD SYSOUT=*
000047 //SYSUT1   DD DSN=HBRLSB0.ISTROBE.INDICENV,DISP=SHR
000048 //SYSUT2   DD DSN=HBRLSB0.ISTROBE.INDICEBK,DISP=SHR
000049 //SYSOUT   DD SYSOUT=*
000050 //SYSIN    DD DUMMY
****** ************************** Bottom of Data **********************
```

Rolf Parker
Systems Programmer (Germany)

Subsystem to influence the allocation of cartridge drives – revisited

The October 2002 issue of *MVS Update* contained an article to influence the allocation of cartridge drives. However, the main program code was not published. It is given below. We apologise for any inconvenience.

```
TITLE 'PPGSSI78 - INFLUENCE ALLOCATION OF CARTRIDGE DEVICES'
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*  THE PURPOSE OF THIS ROUTINE IS TO ENSURE THAT CARTRIDGES WITH A    *
*  VOLUME SERIAL NUMBER LESS THAN 300,000 ARE MOUNTED ON 3490 DEVICE  *
*  TYPES, ESOTERIC NAME OF CART, CARTRIDGES WITH A VOLUME SERIAL      *
*  NUMBER LESS THAN 500,000 ARE MOUNTED ON 3590 DEVICE TYPES,         *
*  ESOTERIC NAME OF STAR, AND ALL OTHERS ARE MOUNTED ON VIRTUAL TAPE  *
*  DEVICE TYPES, ESOTERIC NAME OF VTAPE.                              *
*  IF A USER HAS SPECIFIED AN EXPDT OF 98000 IN HIS JCL, THEN DEVICE  *
*  CONSTRAINTS ARE ASSUMED TO HAVE BEEN SATISFIED.                    *
*  THIS IS A SUBSYSTEM FUNCTION ROUTINE THAT RECEIVES CONTROL IN      *
*  RESPONSE TO A CALL TO SELECT A TAPE DEVICE (SSI FUNCTION CODE 78). *
*  IT MUST BE REENTRANT AND HAVE BEEN LINK-EDITED INTO A LIBRARY      *
*  THAT HAS BEEN INCLUDED IN THE SERIES OF LNKLST CONCATENATIONS.     *
*  ITS INITIALIZATION ROUTINE IS NAMED PPGINT78.                      *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
         SPACE 2
         MACRO
&TAPNAME TAPINFO
         DS    0F
         PUSH PRINT
         PRINT GEN
&TAPNAME DC    CL8'&SYSECT'
         DC    A(&SYSECT)
         DC    CL6'&SYSTIME'
         DC    CL8'&SYSDATE'
         POP  PRINT
         MEND
         EJECT
PPGSSI78 CSECT
         SPACE
PPGSSI78 AMODE 31
PPGSSI78 RMODE 24
         SPACE
         PRINT NOGEN
         SPACE
         USING PPGSSI78,R12        ESTABLISH PPGSSI78 ADDRESSABILITY
         USING PSA,R0              ESTABLISH PSA ADDRESSABILITY
```

```
         SPACE
         BAKR  R14,RØ              PRESERVE ENVIRONMENT AT ENTRY
         LR    R12,R15             PRIME BASE REGISTER
         SPACE
         LR    R1Ø,RØ              PRIME SSCVT BASE
         USING SSCT,R1Ø            ESTABLISH SSCVT ADDRESSABILITY
         SPACE
         LR    R11,R1              PRIME SSOB BASE
         USING SSOB,R11            ESTABLISH SSOB ADDRESSABILITY
         MVC   SSOBRETN,PCPFØ      ASSUME SUCCESSFUL VENTURE
         SPACE
         L     R8,SSOBSSIB         FETCH ADDRESS OF SSIB
         USING SSIB,R8             ESTABLISH SSIB ADDRESSABILITY
         CLC   SSIBSSNM,SSCTSNAM   TEST IF REQUEST SHOULD BE PROCESSED
         BE    PCPVALID            BRANCH IF THIS IS A VALID REQUEST
         SPACE
         MVC   SSOBRETN,PCPF2Ø     INDICATE FORMAT OF SSOB IS INCORRECT
PCPDUST  SR    R15,R15             SET A RETURN CODE OF ZERO
         PR    R14                 BACK TO DUST
         SPACE
         DROP  R8                  FORGET SSIB
         EJECT
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        ASCERTAIN WHICH OF TWO PROCESSING PATHS TO FOLLOW             *
*        BASED ON THE CURRENT JOB'S NAME AND ITS USER'S                *
*        RACF GROUP.                                                   *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
         SPACE
PCPVALID ICM   R7,15,SSOBINDV      GET ADDR OF FUNCTION DEPENDENT AREA
         BE    PCPDUST             BRANCH IF NONE
         USING SSTA,R7             ESTABLISH SSTA ADDRESSABILITY
         SPACE
         CLC   SSTAJNAM,PCPJOBNM   TEST IF THIS IS A TEST JOB
         BNE   PPGNORM             BRANCH IF NOT
         SPACE 2
**********************************************************************
*        ACQUIRE VIRTUAL STORAGE FOR USE AS A WORK AREA               *
**********************************************************************
         SPACE
PGLPRIM  L     RØ,PCPSIZSP         SET SIZE AND SUBPOOL OF STORAGE
         GETMAIN R,LV=(Ø)          GET VIRTUAL STORGE
         SR    RØ,RØ               CLEAR A VOLATILE REGISTER
         IVSK  RØ,R1               FETCH STORAGE
         ST    R13,4(R1)           POINT TO LOWER SAVE AREA
         ST    R1,8(R13)           POINT TO HIGHER SAVE AREA
         LR    R13,R1              SECURE POINTER TO WORK AREA
         USING PCPWORK,R13         ESTABLISH PCPWORK ADDRESSABILITY
         EJECT
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        SET LOOP COUNT BASED ON THE NUMBER OF DDS PRESENT             *
```

```
*        THEN BEGIN PROCESSING EACH DD SECTION                          *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
         SPACE
         ICM   R15,15,SSTANDDS     RETRIEVE NUMBER OF DD SECTIONS
         BE    PCPDPART            BRANCH IF NONE
         SPACE
         L     R14,SSTADDAP        POINT TO THE FIRST DD SECTION
         USING SSTADDA,R14         ESTABLISH SSTADDA ADDRESSABILITY
         SPACE
PCPDODD  MVC   PCPWTO(PCPMSGL),PCPMSG INITIALIZE MESSAGE AREA
         MVC   PCPDDNAM,SSTADDN    STOW NAME OF DD STATEMENT IN WTO
         SPACE
         ICM   RØ,15,SSTANDRA      GET # OF DEVICE REQUEST SECTIONS
         BE    PCPNSECT            BRANCH IF NONE ARE AVAILABLE
         L     R1,SSTADRAP         POINT TO 1ST DEVICE REQUEST SECTION
         USING SSTADRA,R1          ESTABLISH SSTADRA ADDRESSABILITY
         SPACE
         SR    R2,R2               ZERO A VOLATILE GENERAL PURPOSE REG
PCPDEVRQ ICM   R2,3,SSTANDEV       GET # OF ELIGIBLE DEVICE ARRAYS
         BE    PCPNDRAN            BRANCH IF NONE ARE AVAILABLE
         MVC   PCPVOLUM,SSTAVOLI   STOW NUMBER OF VOLUME SERIAL IN WTO
         SPACE
         L     R3,SSTADEVP         POINT TO 1ST ELIGIBLE DEVICE ARRAY
         USING SSTAEDA,R3          ESTABLISH SSTAEDA ADDRESSABILITY
         SPACE
         MVI   PCPAUTH,C'?'        SET USER'S AUTHORIZATION UNKNOWN
         EJECT
**********************************************************************
*        IF USER IS AUTHORIZED, THEN THIS VOLUME WILL BE MOUNTED     *
*        ON A NON-MAGSTAR DEVICE.                                    *
**********************************************************************
         SPACE
         L     R6,PSAAOLD          RETRIEVE ADDRESS OF ASCB
         USING ASCB,R6             ASCB ADDRESSABILITY
         L     R9,ASCBASXB         ADDRESS SPACE EXTENSION BLOCK
         USING ASXB,R9             ASXB ADDRESSABILITY
         DROP  R6                  FORGET ASCB
         L     R6,ASXBSENV         ADDRESS OF ACCESSOR ENVIRONMENT ELE
         USING ACEE,R6             ACEE ADDRESSABILITY
         DROP  R9                  FORGET ASXB
         L     R9,ACEECGRP         CONNECT GROUP DEFINITIONS
         DROP  R6                  FORGET ACEE
         USING CGRP,R9             ESTABLISH CGRP ADDRESSABILITY
         SR    R6,R6               CLEAR WORK REGISTER
         ICM   R6,3,CGRPNUM        FETCH NO. OF ASSOCIATED RACF GROUPS
         BZ    PCPAUSER            ERROR IF NONE
         LA    R9,L'CGRPHADR(R9)   SKIP CGRP HEADER
         DROP  R9                  FORGET CGRP
         SPACE
         MVI   PCPAUTH,C'A'        ASSUME AUTHORIZED USER
```

```
         USING CGRPENTD,R9         ESTABLISH CGRPENTD ADDRESSABILITY
         SPACE
PCPCKID  DS    ØH
         SPACE
         CLC   CGRPNAME(6),=CL6'RACF1' TEST IF GOOD GUYS
         BE    PCPAUSER            BRANCH IF SO
         CLC   CGRPNAME(6),=CL6'RACF2' TEST IF OTHER GOOD GUYS
         BE    PCPAUSER            BRANCH IF SO
         CLC   CGRPNAME(6),=CL6'RACF3' TEST IF COMPUTER OPERATIONS
         BE    PCPAUSER            BRANCH IF SO
         SPACE
         LA    R9,L'CGRPENT(R9)    POINT TO NEXT ENTRY
         BCT   R6,PCPCKID          TRY TO LOCATE CORRESPONDING ENTRY
         SPACE
         MVI   PCPAUTH,C'U'        SHOW UNAUTHORIZED USER
         SPACE
         DROP  R9                  FORGET CGRPENTD
         EJECT
*********************************************************************
*        FORMAT A CARTRIDGE'S EXPIRATION DATE WHENEVER POSSIBLE     *
*********************************************************************
         SPACE
PCPAUSER ICM   R6,15,SSTAJFCP      RETRIEVE ADDRESS OF JFCB
         BE    PCPAITCH            BRANCH IF UNAVAILABLE
         SPACE
         USING PGLJFCBN,R6         ESTABLISH JFCB ADDRESSABILITY
         SPACE
         MVC   PCPDSNAM,JFCBDSNM   COPY DSNAME INTO WTO AREA
         UNPK  PCPMM,JFCBXPDT(2)   ALTER RADIX OF YEAR
         TR    PCPMM(2),PCPTRANS-24Ø CONVERT IT TO EBCDIC
         MVI   PCPMM+2,C' '        REMOVE DE DETRITUS
         UNPK  PCPDDD,JFCBXPDT+1(3) ALTER RADIX OF DAY
         TR    PCPDDD(4),PCPTRANS-24Ø CONVERT IT TO EBCDIC
         MVI   PCPDDD+4,C' '       REMOVE DE DETRITUS
         SPACE 2
*********************************************************************
*        CARTRIDGES WHOSE VOLUME SERIAL NUMBER IS IN THE MAGSTAR    *
*        RANGE, IE 3ØØ,ØØØ - 499,999, WILL BE MOUNTED ON            *
*        NON-MAGSTAR DEVICES WHENEVER A USER IS AUTHORIZED AND      *
*        WHEN HE HAS SPECIFIED AN EXPIRATION DATE OF 98ØØØ TO       *
*        INDICATE THAT THE VOLUME IS AN OUT-OF-HOUSE ONE.           *
*********************************************************************
         SPACE
PCPAITCH CLI   SSTAVOLI,C'9'      TEST IF THIS IS A SCRATCH REQUEST
         BH    PCPSCRAT            BRANCH IF SO
         SPACE
         CLC   JFCBXPDT,PCP98ØØØ   TEST FOR AN OUT-OF-HOUSE CARTRIDGE
         BNE   PCPANOIF            BRANCH IF NOT ONE
         SPACE
         DROP  R6                  FORGET JFCB
```

```
         SPACE
         CLI   PCPAUTH,C'A'        TEST FOR AN AUTHORIZED USER
         BE    PCPPNMAG            BRANCH WHENEVER ONE IS FOUND
         SPACE 2
PCPANOIF CLI   SSTAVOLI,C'3'       TEST IF THIS IS A MAGSTAR CARTRIDGE
         BL    PCPPNMAG            BRANCH IF NOT
         CLI   SSTAVOLI,C'5'       TEST IF THIS IS A "VIRTUAL" TAPE
         BL    PCPDOMAG            BRANCH IF NOT
         EJECT
*********************************************************************
*        PROCESS VIRTUAL CARTRIDGES                                 *
*********************************************************************
         SPACE
PCPDOFOX MVC   PCPUNIT,SSTADNUM    STOW ADDRESS OF TAPE UNIT IN WTO
         MVC   PCPOK,=CL3'YES'     ASSUME DEVICE IS ACCEPTABLE
         CLC   SSTADNUM(2),PGLFOXY TEST IF THIS IS A VIRTUAL DEVICE
         BNE   PCPMARKV            BRANCH IF NOT
         SPACE
PCPETNVT UNPK  PCPMASKS(9),SSTAIBMM(5) STOW IBM MASK IN WTO
         TR    PCPMASKS(8),PCPTRANS-24Ø CONVERT MASK TO EBCDIC
         MVI   PCPMASKS+8,C' '     CLEAR DE DETRITUS
         SPACE
         UNPK  PCPMASKS+9(9),SSTAUSRM(5) STOW USER MASK IN WTO
         TR    PCPMASKS+9(8),PCPTRANS-24Ø CONVERT MASK TO EBCDIC
         MVI   PCPMASKS+8+9,C' '   CLEAR DE DETRITUS
         SPACE
         LA    R3,SSTAEDAL(R3)     NEXT ENTRY IN ELIGIBLE DEVICE ARRAY
         SPACE
         STM   RØ,R15,PCPSAVEA     PRESERVE REGISTERS
         BAS   R9,PCPDISP          ISSUE WTO
         LM    RØ,R15,PCPSAVEA     RESTORE REGISTERS
         SPACE
         BCT   R2,PCPDOFOX         PROCESS NEXT ELIGIBLE DEVICE
         B     PCPNDRAN            PROCESS NEXT DD SECTION
         SPACE
PCPMARKV OI    SSTAUSE1,SSTAINEL   MARK THIS AS AN INELIGIBLE DEVICE
         MVC   PCPOK,=CL3'NO '     SHOW THAT DEVICE IS UNACCEPTABLE
         B     PCPETNVT            PROCESS NEXT ELIGIBLE DEVICE ENTRY
         EJECT
*********************************************************************
*        PROCESS INFORMATION ASSOCIATED WITH SCRATCH REQUESTS       *
*********************************************************************
         SPACE
PCPSCRAT ICM   R6,15,SSTAJFCP      RETRIEVE ADDRESS OF JFCB
         BE    PCPNDRAN            BRANCH IF UNAVAILABLE
         USING PGLJFCBN,R6         ESTABLISH JFCB ADDRESSABILITY
         MVC   PCPDSNAM,JFCBDSNM   COPY DSNAME INTO WTO AREA
         UNPK  PCPMM,JFCBXPDT(2)   ALTER RADIX OF YEAR
         TR    PCPMM(2),PCPTRANS-24Ø CONVERT IT TO EBCDIC
         MVI   PCPMM+2,C' '        REMOVE DE DETRITUS
```

```
         UNPK  PCPDDD,JFCBXPDT+1(3) ALTER RADIX OF DAY
         TR    PCPDDD(4),PCPTRANS-24Ø CONVERT IT TO EBCDIC
         MVI   PCPDDD+4,C' '       REMOVE DE DETRITUS
         MVC   PCPVOLUM,PCPBLNKS   CLEAR PSEUDO VOLUME SERIAL NUMBER
         SPACE
         MVC   PCPOK,=CL3'YES'     SHOW DEVICE IS ALWAYS ACCEPTABLE
         SPACE
         MVC   PCPMASKS,PCPBLNKS   CLEAR MASK AREA
         UNPK  PCPMASKS(13),SSTAVOLI(7) ALTER RADIX OF VOLUME SERIAL
         TR    PCPMASKS(12),PCPTRANS-24Ø CONVERT IT TO EBCDIC
         MVI   PCPMASKS+12,C' '       REMOVE DE DETRITUS
         SPACE
PCPDOSCR MVC   PCPUNIT,SSTADNUM    STOW ADDRESS OF TAPE UNIT IN WTO
         STM   RØ,R15,PCPSAVEA     PRESERVE REGISTERS
         BAS   R9,PCPDISP          ISSUE WTO
         LM    RØ,R15,PCPSAVEA     RESTORE REGISTERS
         SPACE
         LA    R3,SSTAEDAL(R3)     NEXT ENTRY IN ELIGIBLE DEVICE ARRAY
         BCT   R2,PCPDOSCR         PROCESS NEXT ELIGIBLE DEVICE
         B     PCPNDRAN            ONWARD!
         SPACE
         DROP  R6                  FORGET JFCB
         EJECT
*********************************************************************
*        PROCESS NON-MAGSTAR CARTRIDGES                             *
*********************************************************************
         SPACE
PCPPNMAG LA    R4,PGLNMAGS         SET NUMBER OF ENTRIES
         LA    R5,PGLMAGS          POINT TO FIRST ENTRY
         MVC   PCPUNIT,SSTADNUM    STOW ADDRESS OF TAPE UNIT IN WTO
         MVC   PCPOK,=CL3'YES'     ASSUME DEVICE IS ACCEPTABLE
         SPACE
         CLC   SSTADNUM(2),PGLFOXY TEST IF THIS IS A VIRTUAL DEVICE
         BE    PCPMARK             BRANCH IF SO
         SPACE
PCPDOGLC CLC   SSTADNUM(3),Ø(R5)   TEST IF THIS IS A MAGSTAR DEVICE
         BE    PCPMARK             BRANCH IF SO
         SPACE
         LA    R5,3(R5)            POINT TO NEXT ENTRY
         BCT   R4,PCPDOGLC         CHECK NEXT ENTRY
         SPACE
PCPETNNM UNPK  PCPMASKS(9),SSTAIBMM(5) STOW IBM MASK IN WTO
         TR    PCPMASKS(8),PCPTRANS-24Ø CONVERT MASK TO EBCDIC
         MVI   PCPMASKS+8,C' '     CLEAR DE DETRITUS
         UNPK  PCPMASKS+9(9),SSTAUSRM(5) STOW USER MASK IN WTO
         TR    PCPMASKS+9(8),PCPTRANS-24Ø CONVERT MASK TO EBCDIC
         MVI   PCPMASKS+8+9,C' '   CLEAR DE DETRITUS
         LA    R3,SSTAEDAL(R3)     NEXT ENTRY IN ELIGIBLE DEVICE ARRAY
         SPACE
         STM   RØ,R15,PCPSAVEA     PRESERVE REGISTERS
```

```
         BAS   R9,PCPDISP          ISSUE WTO
         LM    RØ,R15,PCPSAVEA     RESTORE REGISTERS
         SPACE
         BCT   R2,PCPPNMAG         PROCESS NEXT ELIGIBLE DEVICE
         SPACE
PCPNDRAN L     R1,SSTADRAN         POINT TO NEXT DEVICE REQUEST
         BCT   RØ,PCPDEVRQ         PROCESS NEXT DEVICE REQUEST
         B     PCPNSECT            PROCESS NEXT DD SECTION
         SPACE
PCPMARK  OI    SSTAUSE1,SSTAINEL   MARK THIS AS AN INELIGIBLE DEVICE
         MVC   PCPOK,=CL3'NO '     SHOW THAT DEVICE IS UNACCEPTABLE
         B     PCPETNNM            PROCESS NEXT ELIGIBLE DEVICE ENTRY
         EJECT
*********************************************************************
*        PROCESS MAGSTAR CARTRIDGES                                 *
*********************************************************************
         SPACE
PCPDOMAG LA    R4,PGLNMAGS         SET NUMBER OF ENTRIES
         LA    R5,PGLMAGS          POINT TO FIRST ENTRY
         MVC   PCPUNIT,SSTADNUM    STOW ADDRESS OF TAPE UNIT IN WTO
         MVC   PCPOK,=CL3'YES'     ASSUME DEVICE IS ACCEPTABLE
         SPACE
         CLC   SSTADNUM(2),PGLFOXY TEST IF THIS IS A VIRTUAL DEVICE
         BE    PCPDING             BRANCH IF SO
         SPACE
PCPDOCPN CLC   SSTADNUM(3),Ø(R5)   TEST IF THIS IS A MAGSTAR DEVICE
         BE    PCPETNLR            BRANCH IF SO
         LA    R5,3(R5)            POINT TO NEXT ENTRY
         BCT   R4,PCPDOCPN         CHECK NEXT ENTRY
         B     PCPDING             EXEMPT THIS DRIVE
         SPACE
PCPETNLR UNPK  PCPMASKS(9),SSTAIBMM(5) STOW IBM MASK IN WTO
         TR    PCPMASKS(8),PCPTRANS-24Ø CONVERT MASK TO EBCDIC
         MVI   PCPMASKS+8,C' '     CLEAR DE DETRITUS
         UNPK  PCPMASKS+9(9),SSTAUSRM(5) STOW USER MASK IN WTO
         TR    PCPMASKS+9(8),PCPTRANS-24Ø CONVERT MASK TO EBCDIC
         MVI   PCPMASKS+8+9,C' '   CLEAR DE DETRITUS
         LA    R3,SSTAEDAL(R3)     NEXT ENTRY IN ELIGIBLE DEVICE ARRAY
         SPACE
         STM   RØ,R15,PCPSAVE      PRESERVE REGISTERS
         BAS   R9,PCPDISP          ISSUE WTO
         LM    RØ,R15,PCPSAVE      RESTORE REGISTERS
         SPACE
         BCT   R2,PCPDOMAG         PROCESS NEXT ELIGIBLE DEVICE
         B     PCPNDRAN            PROCESS NEXT DEVICE REQUEST
         SPACE
PCPDING  OI    SSTAUSE1,SSTAINEL   MARK THIS AS AN INELIGIBLE DEVICE
         MVC   PCPOK,=CL3'NO '     SHOW THAT DEVICE IS UNACCEPTABLE
         B     PCPETNLR            PROCESS NEXT ELIGIBLE DEVICE ENTRY
         SPACE
```

```
PCPNSECT L     R14,SSTADDAN        FETCH ADDRESS OF NEXT DD SECTION
         BCT   R15,PCPDODD         PROCESS IT
         EJECT
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        CLEAN UP AND TERMINATE                                          *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
         SPACE
PCPDPART LA    RØ,PCPWORKL         SET LENGTH OF WORK AREA
         LR    R3,R13              SET ADDRESS OF ACQUIRED STORAGE
         L     RØ,PCPSIZSP         SET SUBPOOL AND SIZE OF AREA TO FREE
         SPACE
         FREEMAIN R,A=(3),LV=(Ø)   RELEASE IT
         SPACE
         SR    R15,R15             SET A RETURN CODE OF ZERO
         PR    R14                 BACK TO DUST
         SPACE
PCPDISP  WTO   MF=(E,PCPWTO)       SHOW RESULTS OF PPGSSI78 PROCESSING
         BR    R9                  RETURN TO CALLER
         EJECT
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        SET LOOP COUNT BASED ON THE NUMBER OF DDS PRESENT               *
*        THEN BEGIN PROCESSING EACH DD SECTION                           *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
         SPACE
PPGNORM  ICM   R15,15,SSTANDDS     RETRIEVE NUMBER OF DD SECTIONS
         BE    PPGDPART            BRANCH IF NONE
         SPACE
         L     R14,SSTADDAP        POINT TO THE FIRST DD SECTION
         USING SSTADDA,R14         ESTABLISH SSTADDA ADDRESSABILITY
         SPACE
PPGDODD  ICM   RØ,15,SSTANDRA      GET # OF DEVICE REQUEST SECTIONS
         BE    PPGNSECT            BRANCH IF NONE ARE AVAILABLE
         L     R1,SSTADRAP         POINT TO 1ST DEVICE REQUEST SECTION
         USING SSTADRA,R1          ESTABLISH SSTADRA ADDRESSABILITY
         SPACE
         SR    R2,R2               ZERO A VOLATILE GENERAL PURPOSE REG
PPGDEVRQ ICM   R2,3,SSTANDEV       GET # OF ELIGIBLE DEVICE ARRAYS
         BE    PPGNDRAN            BRANCH IF NONE ARE AVAILABLE
         SPACE
         L     R3,SSTADEVP         POINT TO 1ST ELIGIBLE DEVICE ARRAY
         USING SSTAEDA,R3          ESTABLISH SSTAEDA ADDRESSABILITY
         SPACE
         CLI   SSTAVOLI,C'9'       TEST IF THIS IS A SCRATCH REQUEST
         BH    PPGNDRAN            BRANCH IF SO
         SPACE
         ICM   R6,15,SSTAJFCP      RETRIEVE ADDRESS OF JFCB
         BE    PPGMPCCA            BRANCH IF UNAVAILABLE
         USING PGLJFCBN,R6         ESTABLISH JFCB ADDRESSABILITY
         SPACE
         CLC   JFCBXPDT,PCP98ØØØ   TEST FOR AN OUT-OF-HOUSE CARTRIDGE
```

```
         BE    PPGNDRAN            BRANCH IS 'TIS
         SPACE
         DROP  R6                  FORGET JFCB
         SPACE
PPGMPCCA CLI   SSTAVOLI,C'3'       TEST IF THIS IS A MAGSTAR CARTRIDGE
         BL    PPGPNMAG            BRANCH IF IMPOSSIBLE
         CLI   SSTAVOLI,C'5'       TEST IF THIS IS A "VIRTUAL" TAPE
         BL    PPGDOMAG            BRANCH IF IMPOSSIBLE
         EJECT
*********************************************************************
*        PROCESS VIRTUAL CARTRIDGES                                 *
*********************************************************************
         SPACE
PPGDOFOX CLC   SSTADNUM(2),PGLFOXY TEST IF THIS IS A VIRTUAL DEVICE
         BNE   PPGMARKV            BRANCH IF NOT
         SPACE
PPGETNVT LA    R3,SSTAEDAL(R3)     NEXT ENTRY IN ELIGIBLE DEVICE ARRAY
         BCT   R2,PPGDOFOX         PROCESS NEXT ELIGIBLE DEVICE
         B     PPGNDRAN            PROCESS NEXT DD SECTION
         SPACE
PPGMARKV OI    SSTAUSE1,SSTAINEL   MARK THIS AS AN INELIGIBLE DEVICE
         B     PPGETNVT            PROCESS NEXT ELIGIBLE DEVICE ENTRY
         SPACE 2
*********************************************************************
*        PROCESS NON-MAGSTAR CARTRIDGES                             *
*********************************************************************
         SPACE
PPGPNMAG CLC   SSTADNUM(2),PGLFOXY TEST IF THIS IS A VIRTUAL DEVICE
         BE    PPGMARK             BRANCH IF SO
         SPACE
         LA    R4,PGLNMAGS         SET NUMBER OF ENTRIES
         LA    R5,PGLMAGS          POINT TO FIRST ENTRY
         SPACE
PPGDOGLC CLC   SSTADNUM(3),Ø(R5)   TEST IF THIS IS A MAGSTAR DEVICE
         BE    PPGMARK             BRANCH IF SO
         LA    R5,3(R5)            POINT TO NEXT ENTRY
         BCT   R4,PPGDOGLC         CHECK NEXT ENTRY
         SPACE
PPGETNNM LA    R3,SSTAEDAL(R3)     NEXT ENTRY IN ELIGIBLE DEVICE ARRAY
         BCT   R2,PPGPNMAG         PROCESS NEXT ELIGIBLE DEVICE
         SPACE
PPGNDRAN L     R1,SSTADRAN         POINT TO NEXT DEVICE REQUEST
         BCT   RØ,PPGDEVRQ         PROCESS NEXT DEVICE REQUEST
         B     PPGNSECT            PROCESS NEXT DD SECTION
         SPACE
PPGMARK  OI    SSTAUSE1,SSTAINEL   MARK THIS AS AN INELIGIBLE DEVICE
         B     PPGETNNM            PROCESS NEXT ELIGIBLE DEVICE ENTRY
         EJECT
*********************************************************************
*        PROCESS MAGSTAR CARTRIDGES                                 *
```

```
*********************************************************************
         SPACE
PPGDOMAG CLC   SSTADNUM(2),PGLFOXY TEST IF THIS IS A VIRTUAL DEVICE
         BE    PPGDING             BRANCH IF SO
         SPACE
         LA    R4,PGLNMAGS         SET NUMBER OF ENTRIES
         LA    R5,PGLMAGS          POINT TO FIRST ENTRY
         SPACE
PPGDOCPN CLC   SSTADNUM(3),Ø(R5)   TEST IF THIS IS A MAGSTAR DEVICE
         BE    PPGETNLR            BRANCH IF SO
         LA    R5,3(R5)            POINT TO NEXT ENTRY
         BCT   R4,PPGDOCPN         CHECK NEXT ENTRY
         B     PPGDING             EXEMPT THIS DRIVE
         SPACE
PPGETNLR LA    R3,SSTAEDAL(R3)     NEXT ENTRY IN ELIGIBLE DEVICE ARRAY
         BCT   R2,PPGDOMAG         PROCESS NEXT ELIGIBLE DEVICE
         B     PPGNDRAN            PROCESS NEXT DEVICE REQUEST
         SPACE
PPGDING  OI    SSTAUSE1,SSTAINEL   MARK THIS AS AN INELIGIBLE DEVICE
         B     PPGETNLR            PROCESS NEXT ELIGIBLE DEVICE ENTRY
         SPACE 3
PPGNSECT L     R14,SSTADDAN        FETCH ADDRESS OF NEXT DD SECTION
         BCT   R15,PPGDODD         PROCESS IT
         SPACE
PPGDPART SR    R15,R15             SET A RETURN CODE OF ZERO
         PR    R14                 BACK TO DUST
         EJECT
         EJECT
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        CONSTANTS, EQUATES, AND OTHER SUCH NONSENSE                  *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
         SPACE
PCPFØ    DC    F'Ø'
PCPF2Ø   DC    F'2Ø'
PCPJOBNM DC    CL8'XEPHON'
PCPTRANS DC    C'Ø123456789ABCDEF'
PCP98ØØØ DC    XL3'62ØØØØ'
PCPBLNKS DC    CL18' '
         SPACE 3
         DS    ØF
PCPMSG   DC    AL2(PCPMSGL)
         DC    XL2'8ØØØ'
         DC    CL8' '
         DC    CL2' '
         DC    CL6' '
         DC    CL2' '
         DC    CL4' '
         DC    CL2' '
         DC    CL3' '
         DC    C' '
```

```
         DC    CL18' '
         DC    CL2' '
         DC    CL3' '
         DC    CL5' '
         DC    CL44' '
         DC    XL2'0000'
         DC    XL2'0020'
         SPACE
PCPMSGL  EQU   (*-PCPMSG)
         EJECT
PCPSIZSP DC    0F'0',AL1(230),AL3(PCPWORKL)
         SPACE
PGLFOXY  DC    CL2'0F'
PGLMAGS  DC    CL3'047'
         DC    CL3'048'
         DC    CL3'049'
PGLNMAGS EQU   ((*-PGLMAGS)/L'PGLMAGS)
         SPACE
         TAPINFO
         SPACE
         YREGS
         SPACE 2
         LTORG
         EJECT
PCPWORK  DSECT
PCPSAVE  DS    18F
PCPSAVEA DS    16F
         SPACE
PCPWTO   DC    AL2(PCPMSGL)
         DC    XL2'8000'
PCPDDNAM DC    CL8' '
         DC    CL2' '
PCPVOLUM DC    CL6' '
         DC    CL2' '
PCPUNIT  DC    CL4' '
         DC    CL2' '
PCPOK    DC    CL3' '
         DC    C' '
PCPMASKS DC    CL18' '
PCPAUTH  DC    C' '
         DC    C' '
PCPMM    DC    CL3' '
PCPDDD   DC    CL5' '
PCPDSNAM DC    CL44' '
         DC    XL2'0000'
         DC    XL2'0020'
         DS    0D
PCPWORKL EQU   *-PCPWORK
         TITLE 'GENERATE OS/390 CONTROL BLOCKS'
         ICHPCGRP
```

```
         SPACE
         IHAACEE
         SPACE
         IHAASXB
         SPACE
         IHAASCB
         SPACE
         IEFJSCVT
         SPACE
         IHAPSA
         SPACE
         IEFJSSIB
         SPACE
         IEFSSOBH
         SPACE
         IEFSSTA
         SPACE
PGLJFCBN DSECT
         IEFJFCBN LIST=YES
         SPACE
         END
```


Back-ups and offsite recovery – part 2

This month we conclude the code for a system to carry out full-pack back-ups for offsite recovery, which also handles the restores at the recovery site.

HDRDATE

```
/* ---------------------------------------------------- */
/* REXX "HDRDATE"                                        */
/* Write out date record, for bottom of BKUPSITE listings    */
/* ---------------------------------------------------- */
Address "TSO"
xx1 = "-      >>>>>>>>>>>>>>>>>>>>>>>> Run: "Date()
xx1 = Left(xx1" <<<<<<<<<<<<<<<<<<<<<<<<          ",81)
"EXECIO 1 DISKW HDRDT (Stem xx"
Return
This is "LASTGEN":
PROC 1 GDG
/* ============================================================== */
```

```
/*  GENFIND - FIND LATEST GENERATION OF A GDG AND PASS BACK AS A    */
/*            RETURN CODE. THIS WILL ALLOW EXPLICIT ALLOCATION OF   */
/*            THE NEXT GENERATION (AS OPPOSED TO '+1').             */
/*  INPUT   - GDG - THE NAME OF THE GDG BASE INVOLVED.              */
/*  NB      - CONTROL MUST HAVE 'LIST' AND 'MSG' SPECIFIED (TO      */
/*            BE ABLE TO TRAP OUTPUT FROM 'LISTC' COMMAND).         */
/*  OUTPUT  - RETURN CODES:                                         */
/*            55555  -  NO GENERATIONS EXIST                        */
/*            99999  -  NAME PASSED WAS NOT A GDG BASE              */
/*            OTHER  -  NUMERIC VALUE OF LAST GENERATION FOUND FOR  */
/*                      GDG BASE (IE 12 FOR 'TEST.GDG.G00012V00').  */
/* ============================================================== */
       CONTROL FLUSH LIST NOSYMLIST NOCONLIST MSG NOPROMPT
/*     CONTROL MAIN FLUSH LIST SYMLIST CONLIST MSG PROMPT
/* ============================================================== */
/*  DO A 'LISTC' FOR SPECIFIED GDG BASE AND TRAP OUTPUT IN THE     */
/*  &SYSOUTLINE ARRAY. IF NO GDG BASE ENTRY FOUND, EXIT...         */
/* ============================================================== */
       SET &SYSOUTTRAP = 500
       LISTC ENT('&GDG') NAMES
       SET A = &STR(&SYSOUTLINE2)
       SET X = &SUBSTR(1:8,&STR(&A))
       IF &STR(&X) ¬= &STR(GDG BASE) THEN DO
           WRITE ENTRY '&GDG' IS INVALID OR NOT A GDG BASE...
           EXIT CODE(99999)
       END
/* ============================================================== */
/*  LOOP THRU THE 'LISTC' OUTPUT. SAVE ANY (NONVSAM) ASSOCIATIONS  */
/*  FOUND IN &LASTGEN, SO THAT AT THE END &LASTGEN WILL CONTAIN    */
/*  THE 'REAL' LATEST GENERATION...                                */
/* ============================================================== */
       SET A = 1
       SET B = &SYSOUTLINE
       SET LASTGEN = NONE
       DO WHILE &A <= &B
           SET C = &STR(&&SYSOUTLINE&A)
           SET X = &SUBSTR(1:14,&STR(&C))
           IF &STR(&X) = &STR(   NONVSAM --) THEN DO
               SET Y = &LENGTH(&STR(&C)) - 7
               SET Z = &Y + 7
               SET LASTGEN = &SUBSTR(&Y:&Z,&STR(&C))
           END
           SET &A = &A + 1
       END
       IF &LASTGEN = NONE THEN DO
           WRITE NO GENERATIONS FOR '&GDG'...
           EXIT CODE(55555)
       END
/* ============================================================== */
/*  WE NOW HAVE 'GXXXXV00' - GET THE XXXX AND USE AS RETURN CODE... */
```

```
/* ============================================================= */
          SET &CODE = &SUBSTR(2:5,&LASTGEN)
          EXIT CODE(&CODE)
```

OPRMSG3

```
/* ---------------------------------------------------------- */
/* REXX "OPRMSG3"                                             */
/* Send msg to user who subbed failing disk initialize job    */
/* ---------------------------------------------------------- */
Parse Upper Arg user cuu vol .
ADDRESS "TSO"
"SEND '==> Job "user"I (initialize "cuu" as "vol") has failed. <==',
      U("user")"
Return
```

OPRMSG4

```
/* ---------------------------------------------------------- */
/* REXX "OPRMSG4"                                             */
/* Send msg to the Ops (reports job may have failed)          */
/* ---------------------------------------------------------- */
Parse Upper Arg user step .
ADDRESS "TSO"
"SEND '==> Job "user"P, Step "step" had RC>Ø. Please check.',
      U("user")"
Return
```

PRTMEMS

```
/* ---------------------------------------------------------- */
/* REXX "PRTMEMS"                                             */
/* Print the contents of the back-up suites, along with the   */
/* jobname of the associated restore job.                     */
/* ---------------------------------------------------------- */
/* ---------------------------------------------------------- */
/* Parms passed:                                              */
/*   DSN  Dataset name to be printed (specified as DSN=...)   */
/* ---------------------------------------------------------- */
Parse upper arg dsn .
If Left(dsn,4) ¬= "DSN=" Then Do
  Say " ==> DSN= not specified for first parm..."
  Exit 8
End
a = Length(dsn) - 4
dsname = Right(dsn,a)
jobnm = "RSMVS"
/* ---------------------------------------------------------- */
/* List all members from the dataset using "LISTDS". This     */
```

```
/* will also tell us if the datasets exists, or if we've    */
/* been passed an invalid dataset name...                   */
/* -------------------------------------------------------- */
dsn = "'"dsname"'"
Address "TSO"
x = outtrap("out.",2000,noconcat)
"LISTDS "dsn" MEMBERS"
lastcc = rc
x = outtrap("OFF")
If lastcc = 12 Then Do
  Say " ********************************** "
  Say " * Invalid dataset name was passed * "
  Say " ********************************** "
  Exit 8
End
If Left(out.2,9) = "IKJ585Ø3I" Then Do
  Say " ************************************** "
  Say " * The specified dataset does not exist * "
  Say " ************************************** "
  Exit 8
End
"DROPBUF"
members_printed = Ø
Do a = 7 to out.Ø                /* First 6 lines contain headings */
  mbr = Strip(out.a)
  Call PRINT_IT
  members_printed = members_printed + 1
End
members_printed = Strip(members_printed)
Say " "
Say " ********************************** "
a = " * Number of members printed = "members_printed
a = Left(a"      ",36)
a = a"*"
Say a
Say " ********************************** "
Say " "
"DROPBUF"
qnum = Queued()
"EXECIO "qnum" DISKW SYSUT2 (FINIS"
"FREE FI(SYSUT2)"
Return
/* -------------------------------------------------------- */
/* Allocate the member and read it in...                    */
/* -------------------------------------------------------- */
PRINT_IT:
"DROPBUF"                        /* Clear out stack              */
Say " Printing member: "mbr
Queue "1 "                       /* Start new page               */
name = dsname"("mbr")"           /* Heading                      */
head = Left(" -------------------- "name" --------------------",8Ø)
```

```
Queue head
Call READ_MEMBER                  /* Read in member                */
qnum = Queued()
"EXECIO "qnum" DISKW SYSUT2"      /* Write out break/heading       */
Return
/* ---------------------------------------------------- */
/*  Allocate the specified member and read into the stack   */
/* ---------------------------------------------------- */
READ_MEMBER:
thename = "('"dsname"("mbr")') SHR"
"ALLOC FI(SYSUT1) DA"thename
"EXECIO * DISKR SYSUT1 (Stem data. FINIS"
"FREE FI(SYSUT1)"
bit = Right(mbr,1)                 /* Get suite suffix              */
Do xx = 1 to data.Ø
  restnam = jobnm||bit||Right("Ø"xx,2) /* Restore jobnames at C'disco*/
  If bit = "O" Then                    /* Different hdng for BCKMVSO */
    restnam = "!!To be restored standalone!!"
  thedata = " "Left(data.xx,3Ø)||restnam
  Queue thedata                    /* Queue lines/force double space */
End
Return
```

ISPF SKELETONS

SOPBK170

```
)CM ********************************************************************
)CM * +++ BACK-UP VOLUMES FOR OFFSITE RECOVERY...                 +++ *
)CM ********************************************************************
//&BKJBNM JOB ,'&SUITE BKUP &DISVOL',NOTIFY=&USRID,CLASS=5,
//    MSGLEVEL=(1,1),MSGCLASS=E,REGION=8M
//*
//* BACKUP &DISVOL ( &DISADR )
//*
//&DISVOL  EXEC PGM=ADRDSSU
//STEPLIB  DD  DISP=SHR,DSN=BQIBI.OPSREC.ADRDSSU
//SYSPRINT DD  SYSOUT=*
//DASD1    DD  DISP=SHR,UNIT=&DISDEV,VOL=SER=&DISVOL
//TAPE1    DD  DSN=SYS8.&SYS..&SUITE..&DISVOL(+1),
//             DISP=(NEW,CATLG,DELETE),LABEL=EXPDT=99ØØØ,
//             UNIT=ROBOØØ,DCB=SYS3.NULL.PATT
//SYSIN   DD *
 DUMP FULL INDDNAME(DASD1) OUTDDNAME(TAPE1) WAIT(Ø,Ø) -
   ADMIN OPTIMIZE(4)
/*
//         IF  RC GT Ø THEN
//S2      EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM &SUITE &DISVOL'
```

```
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
//         IF  ABEND THEN
//S3      EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM &SUITE &DISVOL'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
```

SOPBK171

```
)CM ***************************************************************
)CM * +++ RESTORE A VOLUME ONSITE...                          +++ *
)CM ***************************************************************
//&BKJBNM JOB ,'&SUITE : &DISVOL',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=V
//*
//* RESTORE &DISVOL ONSITE
//*
//S1      EXEC PGM=ADRDSSU,REGION=4M
//SYSPRINT DD  SYSOUT=*
//DASD1    DD  DISP=SHR,UNIT=/&BTGT,VOL=SER=&BTGTVL
//BACKUP1  DD  DSN=SYS8.&SYS..&SUITE..&DISVOL(&BGN),
//             DISP=SHR
//SYSIN   DD *
 RESTORE FULL INDDNAME(BACKUP1) OUTDDNAME(DASD1) WAIT(Ø,Ø) COPYVOLID
/*
//         IF  RC GT 4 THEN
//S2      EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM &SUITE &DISVOL'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
//         IF  ABEND THEN
//S3      EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM &SUITE &DISVOL'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
```

SOPBK172

```
)CM ***************************************************************
```

```
)CM * +++ REFRESH THE BACK-UP CATALOGS AND DATASETS ON "ONEPAC", +++ *
)CM * +++ THEN BACK UP "ONEPAC" (TO BE SENT TO BKUPSITE).        +++ *
)CM ****************************************************************
//&BKJBNM JOB ,'NOVAMDS',CLASS=L,REGION=8M,
//        MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&USRID
//*--------------------------------------------------------------
//* STEPØ  - COMPRESS SYS2.PROCLIB ON "ONEPACK" SYSTEM
//*        - RECREATE MEMBER "WHEN" ON "ONEPACK" SYSTEM
//*--------------------------------------------------------------
//STEPØ    EXEC PGM=IEBCOPY
//SYSPRINT  DD SYSOUT=*
//IN2      DD DISP=SHR,DSN=SYS1.PROCLIB,UNIT=339Ø,VOL=SER=ONEPAC
//OU2      DD DISP=SHR,DSN=SYS1.PROCLIB,UNIT=339Ø,VOL=SER=ONEPAC
//SYSIN    DD *
 COPY INDD=((IN2,R)),OUTDD=OU2
/*
//SØØ    EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DATA,DLM='ŁŁ'
//WHEN   PROC
//OPSMSG EXEC PGM=IPLMSG,PARM='+++ ONEPAC BUILD DATE: &BLDDAT +++'
ŁŁ
//SYSUT2   DD DISP=SHR,DSN=SYS1.PROCLIB(WHEN),UNIT=339Ø,VOL=SER=ONEPAC
//SYSIN    DD DUMMY
//*--------------------------------------------------------------
//* STEPØ1 - DELETE/DEFINE BACKUP UCATS ON "ONEPACK" SYSTEM
//*--------------------------------------------------------------
//SØ1    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DEL (CATALOG.DRBACKUP.ONEPAC.SYS1) UCAT RECOVERY
 DEL (CATALOG.DRBACKUP.ONEPAC.SYS2) UCAT RECOVERY
 DEL (CATALOG.DRBACKUP.ONEPAC.SYS3) UCAT RECOVERY
 SET MAXCC = Ø
 DEF UCAT (NAME(CATALOG.DRBACKUP.ONEPAC.SYS1) -
  VOLUMES (ONEPAC) CYL(5 1) IMBED REPLICATE WRITECHECK ICFCATALOG) -
      DATA(CISZ(1Ø24)) -
      INDEX(CISZ(1Ø24))
 DEF UCAT (NAME(CATALOG.DRBACKUP.ONEPAC.SYS2) -
  VOLUMES (ONEPAC) CYL(5 1) IMBED REPLICATE WRITECHECK ICFCATALOG) -
      DATA(CISZ(1Ø24)) -
      INDEX(CISZ(1Ø24))
 DEF UCAT (NAME(CATALOG.DRBACKUP.ONEPAC.SYS3) -
  VOLUMES (ONEPAC) CYL(5 1) IMBED REPLICATE WRITECHECK ICFCATALOG) -
      DATA(CISZ(1Ø24)) -
      INDEX(CISZ(1Ø24))
/*
//*--------------------------------------------------------------
//* STEPØ2 - COPY ALL "SYS8" ENTRIES INTO BACK-UP UCAT
//*--------------------------------------------------------------
```

```
//SØ2    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//DDSYS1I  DD DSN=CATALOG.DRBACKUP.SYS1,DISP=SHR
//DDSYS1O  DD DSN=CATALOG.DRBACKUP.ONEPAC.SYS1,DISP=OLD  <<DISP=OLD>>
//DDSYS2I  DD DSN=CATALOG.DRBACKUP.SYS2,DISP=SHR
//DDSYS2O  DD DSN=CATALOG.DRBACKUP.ONEPAC.SYS2,DISP=OLD  <<DISP=OLD>>
//DDSYS3I  DD DSN=CATALOG.DRBACKUP.SYS3,DISP=SHR
//DDSYS3O  DD DSN=CATALOG.DRBACKUP.ONEPAC.SYS3,DISP=OLD  <<DISP=OLD>>
//SYSIN    DD *
 REPRO INFILE(DDSYS1I) OUTFILE(DDSYS1O) REPLACE
 REPRO INFILE(DDSYS2I) OUTFILE(DDSYS2O) REPLACE
 REPRO INFILE(DDSYS3I) OUTFILE(DDSYS3O) REPLACE
/*
//*-------------------------------------------------------------
//* STEPØ5 - USE "SUPERSCR" TO SCRATCH THE EXISTING "BQIBI.OPSREC"
//*          DATASETS FROM THE "ONEPAC" VOLUME.
//*-------------------------------------------------------------
//SØ5A     EXEC  PGM=SUPERSCR
//SCRATCH  DD DISP=SHR,VOL=SER=ONEPAC,DSN=X.BQIBIØ6.BKPS.BACKUPS,
//         UNIT=339Ø
//SYSPRINT DD SYSOUT=*
//SØ5B     EXEC  PGM=SUPERSCR
//SCRATCH  DD DISP=SHR,VOL=SER=ONEPAC,DSN=X.BQIBI.OPSREC.MSGS,
//         UNIT=339Ø
//SYSPRINT DD SYSOUT=*
//SØ5C     EXEC  PGM=SUPERSCR
//SCRATCH  DD DISP=SHR,VOL=SER=ONEPAC,DSN=X.BQIBI.OPSREC.PANELS,
//         UNIT=339Ø
//SYSPRINT DD SYSOUT=*
//SØ5D     EXEC  PGM=SUPERSCR
//SCRATCH  DD DISP=SHR,VOL=SER=ONEPAC,DSN=X.BQIBI.OPSREC.REXX,
//         UNIT=339Ø
//SYSPRINT DD SYSOUT=*
//SØ5E     EXEC  PGM=SUPERSCR
//SCRATCH  DD DISP=SHR,VOL=SER=ONEPAC,DSN=X.BQIBI.OPSREC.SKELS,
//         UNIT=339Ø
//SYSPRINT DD SYSOUT=*
//*-------------------------------------------------------------
//* STEPØ6 - COPY THE "BQIBI.OPSREC" DATASETS TO "ONEPAC"
//*-------------------------------------------------------------
//SØ6    EXEC PGM=ADRDSSU,REGION=4Ø96K,PARM='UTILMSG=YES'
//SYSPRINT DD SYSOUT=*
//OUTDASD  DD DISP=SHR,UNIT=339Ø,VOL=SER=ONEPAC
//SYSIN    DD *
 COPY OUTDD(OUTDASD) TOL(ENQF) REPLACE SHARE -
 DATASET(INCLUDE(BQIBIØ6.BKPS.BACKUPS,   -
                 BQIBIØ6.BKPS.MSGS,      -
                 BQIBIØ6.BKPS.PANELS,    -
                 BQIBIØ6.BKPS.REXX,      -
                 BQIBIØ6.BKPS.SKELS))
```

```
 COPY OUTDD(OUTDASD) TOL(ENQF) REPLACE SHARE
//*-------------------------------------------------------------
//* STEPØ8 - BACK UP THE RECREATED "ONEPAC" SYSTEM
//*-------------------------------------------------------------
//SØ8     EXEC PGM=ADRDSSU
//SYSPRINT DD  SYSOUT=*
//DASD1    DD  DISP=SHR,UNIT=/4C4,VOL=SER=ONEPAC
//TAPE1    DD  DSN=SYS8.SYS1.BCKMVSO.ONEPAC(+1),
//             DISP=(NEW,CATLG,DELETE),LABEL=EXPDT=99ØØØ,
//             UNIT=ROBOØØ,DCB=SYS3.NULL.PATT
//SYSIN   DD *
 DUMP FULL INDDNAME(DASD1) OUTDDNAME(TAPE1)           WAIT(Ø,Ø) -
   ADMIN OPTIMIZE(4)
/*
//         IF  RC GT 4 THEN
//SØ8B    EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM BCKMVSO ONEPAC'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
//         IF  ABEND THEN
//SØ8C    EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM BCKMVSO ONEPAC'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
//
```

SOPBK173

```
)CM ***************************************************************
)CM * +++ RESTORE VOLUMES AT BKUPSITE...                    +++ *
)CM ***************************************************************
//&BKJBNM JOB ,'&SUITE &DISVOL &COMUNIT',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=A
//*
//* (BKUPSITE) RESTORE VOLUME "&DISVOL"
//*
//S1      EXEC PGM=ADRDSSU,REGION=4M
//SYSPRINT DD  SYSOUT=*
//DASD1    DD  DISP=SHR,UNIT=/&COMUNIT,VOL=SER=&COMVOL
//BACKUP1  DD  DSN=SYS8.&SYS..&SUITE..&DISVOL(&BGN),
//             DISP=SHR
//SYSIN   DD *
 RESTORE FULL INDDNAME(BACKUP1) OUTDDNAME(DASD1) WAIT(Ø,Ø) COPYVOLID
/*
//         IF  RC GT 4 THEN
//S2      EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM &SUITE &DISVOL'
```

```
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
//         IF  ABEND THEN
//S3      EXEC PGM=IKJEFTØ1,PARM='%DRMSG1 &BKJBNM &SUITE &DISVOL'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.REXX
//        ENDIF
```

SOPBK174

```
)CM ************************************************************
)CM * +++ INITIALIZE A VOLUME AT BKUPSITE (INSTEAD OF A RESTORE, +++ *
)CM * +++ EG 'PSTR' VOLUMES').                                   +++ *
)CM ************************************************************
//&BKJBNM JOB ,'INIT. &SUITE &DISVOL',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=S
//*-------------------------------------------***
//* (BKUPSITE) JUST INITIALIZE VOLUME "&DISVOL"   ***
//*-------------------------------------------***
//* INITIALIZE THE VOLUME. MUST BE OFFLINE FIRST ***
//*-------------------------------------------***
//VARY1   EXEC PGM=MVSCMD,PARM='V &COMUNIT,OFFLINE'
//WAIT    EXEC PGM=WAIT,PARM='2'
//INIT1   EXEC PGM=ICKDSF,REGION=1Ø24K
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
 INIT UNITADDRESS(&COMUNIT) -
      VTOC(Ø,1,14) -
      INDEX(1,Ø,15) -
      OWNERID(PROD) -
      VOLID(&DISVOL) -
      VFY(&COMVOL) -
      PURGE
//         IF  RC GT Ø THEN
//S2      EXEC PGM=IKJEFTØ1,PARM='%OPRMSG3 &USRID &COMUNIT &DISVOL'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.BKPS.REXX
//        ENDIF
//         IF  ABEND THEN
//S3      EXEC PGM=IKJEFTØ1,PARM='%OPRMSG3 &USRID &COMUNIT &DISVOL'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
```

```
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.BKPS.REXX
//        ENDIF
```

SOPBK175

```
)CM ********************************************************************
)CM * +++ GENERATE THE LISTINGS FOR BKUPSITE. THE ARE:           +++ *
)CM * +++ A) A HEADER                                           +++ *
)CM * +++ B) CARTS BY BACK-UP SUITE                             +++ *
)CM * +++ C) ALL CARTS                                         +++ *
)CM * +++ D) A DISK MAP                                        +++ *
)CM * +++ E) DASD REQUIRED FOR SYS1, SYS3 AND SYS2             +++ *
)CM * +++ F) ALL DASD REQUIRED                                 +++ *
)CM * +++ G) A LISTING OF ALL BACK-UP SUITES                   +++ *
)CM * +++                                                     +++ *
)CM * +++ ROUTINES CALLED:                                     +++ *
)CM * +++                                                     +++ *
)CM * +++    FAILMAST    PRODUCE A LIST OF ALL DISKS REQUIRED,  +++ *
)CM * +++                BROKEN DOWN INTO SYS1, SYS2, SYS3, AND +++ *
)CM * +++                ALSO A COMPLETE LIST. THE LISTS WILL BE +++ *
)CM * +++                PRINTED HERE AND ALSO E-MAILED DIRECT  +++ *
)CM * +++                TO BKUPSITE.                           +++ *
)CM * +++                                                      +++ *
)CM * +++    PRTMEMS     PRODUCES A LIST OF THE CONTENTS OF A   +++ *
)CM * +++                BACK-UP SUITE MEMBER, ALONG WITH THE   +++ *
)CM * +++                JOBNAME THAT WILL BE SUBMITTED TO BACK +++ *
)CM * +++                IT UP. THIS CAN BE USED AS A 'RUNSHEET'.+++ *
)CM * +++                                                      +++ *
)CM * +++    HDRDATE     PRODUCES A LINE CONTAINING THE DATE THE +++ *
)CM * +++                REPORTS ARE PRODUCED (CONCATENATED ONTO +++ *
)CM * +++                THE BOTTOM OF THE HEADER).             +++ *
)CM * +++                                                      +++ *
)CM * +++    GENLIST1    GROUP OF REXX EXECS THAT EXTRACT THE   +++ *
)CM * +++    GENLIST2    LATEST CARTRIDGE INFORMATION AND THEN  +++ *
)CM * +++    GENLIST3    PRODUCE A 1Ø-ACROSS-THE-PAGE LIST OF   +++ *
)CM * +++                ALL CARTRIDGES REQUIRED FOR BKUPSITE,  +++ *
)CM * +++                AND A JOB TO EJECT THEM.               +++ *
)CM ********************************************************************
//&BKJBNM JOB ,'BKUPSITE PRINT',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=A
//*--------------- GET LIST OF SYS8 GDGS -------------------------
//LISTC1  EXEC PGM=IDCAMS
//SYSPRINT DD  DISP=(,PASS),SPACE=(CYL,(1Ø,1Ø)),DSN=&&LC1,UNIT=SYSDA
//SYSIN    DD  *
 LISTC LVL(SYS8) GDG
/*
//*--------------- EXTRACT OUT THE GDG BASE NAMES ----------------
//RUNREXX1 EXEC PGM=IKJEFTØ1,DYNAMNBR=2ØØ,
//           PARM=('%GENLIST1')
```

```
//SYSTSIN  DD  DUMMY
//LISTING  DD  DISP=(OLD,PASS),DSN=&&LC1
//OUTPUT   DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&NAMES,UNIT=SYSDA
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DSN=BQIBIØ6.BKPS.REXX,DISP=SHR
//SYSPROC  DD  DSN=SYS3.CLIB,DISP=SHR   <== CONTAINS 'LASTGEN'
//*---------------- GET LIST OF SYS8 GDGS -------------------------
//LISTC2   EXEC PGM=IDCAMS
//SYSPRINT DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&LC2,UNIT=SYSDA
//SYSIN    DD  DISP=(OLD,PASS),DSN=&&NAMES
//*---------------- PRODUCE 2 MORE LISTINGS -----------------------
//RUNREXX2 EXEC PGM=IKJEFTØ1,DYNAMNBR=2ØØ,
//           PARM=('%GENLIST2')
//SYSTSIN  DD  DUMMY
//LISTING  DD  DISP=(OLD,PASS),DSN=&&LC2
//OUTPUT   DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&BYBKP,UNIT=SYSDA
//VOLIDS   DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&VOLID,UNIT=SYSDA
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DSN=BQIBIØ6.BKPS.REXX,DISP=SHR
//*---------------- SORT VOLIDS INTO ORDER ------------------------
//SORT1    EXEC PGM=SORT,REGION=2Ø48K
//SORTIN   DD  DISP=(OLD,PASS),DSN=&&VOLID
//SORTOUT  DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&VOLSRT,UNIT=SYSDA
//SORTLIB  DD  DISP=SHR,DSNAME=SYS1.SORTLIB
//SORTWKØ1 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWKØ2 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWKØ3 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SYSOUT   DD  SYSOUT=*
//SYSIN    DD  *
  SORT FIELDS=(4,6,CH,A)
/*
//*---------------- PRODUCE LIST OF ALL CARTS ---------------------
//*-------------- COMPRESS EJECT DATASET FIRST --------------------
//COMPRESS EXEC PGM=IEBCOPY
//SYSPRINT DD  SYSOUT=*
//IN2      DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.EJECT
//OU2      DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.EJECT
//SYSIN    DD  *
 COPY INDD=((IN2,R)),OUTDD=OU2
/*
//RUNREXX3 EXEC PGM=IKJEFTØ1,DYNAMNBR=2ØØ,
//           PARM=('%GENLIST3 ALL')
//SYSTSIN  DD  DUMMY
//VOLIDS   DD  DISP=(OLD,PASS),DSN=&&VOLSRT
//OUTPUT   DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&CARTS,UNIT=SYSDA
//EJECTS   DD  DISP=SHR,DSN=BQIBIØ6.OPSREC.EJECT(ALL)
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DSN=BQIBIØ6.BKPS.REXX,DISP=SHR
//*-------- SORT COMDBKPS INTO ADDRESS ORDER ---------------------
//SORT2   EXEC PGM=SORT,REGION=2Ø48K
```

```
//SORTIN   DD  DISP=SHR,DSN=BQIBI.OPSREC.BACKUPS(COMDBKPS)
//SORTOUT  DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&COMD1,UNIT=SYSDA
//SORTLIB  DD  DISP=SHR,DSNAME=SYS1.SORTLIB
//SORTWKØ1 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWKØ2 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWKØ3 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SYSOUT   DD  SYSOUT=*
//SYSIN    DD  *
  SORT FIELDS=(36,5,CH,A)
/*
//*-------- SORT COMDBKPS INTO ADDRESS WITHIN SUITE ORDER ---------
//SORT3   EXEC PGM=SORT,REGION=2Ø48K
//SORTIN   DD  DISP=SHR,DSN=BQIBI.OPSREC.BACKUPS(COMDBKPS)
//SORTOUT  DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&COMD2,UNIT=SYSDA
//SORTLIB  DD  DISP=SHR,DSNAME=SYS1.SORTLIB
//SORTWKØ1 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWKØ2 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWKØ3 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SYSOUT   DD  SYSOUT=*
//SYSIN    DD  *
  SORT FIELDS=(19,7,CH,A,36,5,CH,A)
/*
//*------------------ CREATE THE DASD REPORTS ----------------------
//RUNREXX4 EXEC PGM=IKJEFTØ1,DYNAMNBR=2ØØ,
//            PARM=('%FAILMAST')
//SYSTSIN  DD  DUMMY
//COMDBKP1 DD  DISP=(OLD,PASS),DSN=&&COMD1
//COMDBKP2 DD  DISP=(OLD,PASS),DSN=&&COMD2
//SYSTSPRT DD  SYSOUT=*
//DASDLSTF DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&SYS1,UNIT=SYSDA
//DASDLSTC DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&SYS3,UNIT=SYSDA
//DASDLSTP DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&SYS2,UNIT=SYSDA
//DASDALL  DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&DASD,UNIT=SYSDA
//SYSEXEC  DD  DSN=BQIBIØ6.BKPS.REXX,DISP=SHR
//*
//         IF  RC GT Ø THEN
//OPSMSG  EXEC PGM=IKJEFTØ1,PARM='%OPRMSG4 &USRID RUNREXX4'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.BKPS.REXX
//        ENDIF
//*
//RUNREXX5 EXEC PGM=IKJEFTØ1,DYNAMNBR=5Ø,
// PARM='%PRTMEMS DSN=BQIBI.OPSREC.BACKUPS'
//SYSPRINT DD  SYSOUT=*
//SYSUT2   DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&SUITES,UNIT=SYSDA
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.BKPS.REXX
//SYSTSIN  DD  DUMMY
```

```
//*
//RUNREXX6 EXEC PGM=IKJEFTØ1,DYNAMNBR=2Ø,
//            PARM=('%HDRDATE')
//SYSTSIN  DD  DUMMY
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DISP=SHR,DSN=BQIBIØ6.BKPS.REXX
//SYSTSIN  DD  DUMMY
//HDRDT    DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&HDRDT,UNIT=SYSDA,
//         LRECL=81,RECFM=FBA,BLKSIZE=81ØØ
//*--------------------- PRINT THE REPORTS ----------------------
//GENER1   EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=SHR,DSN=BQIBI.OPSREC.LISTING(XERPORT) SET PORTRAIT
//         DD  DISP=SHR,DSN=BQIBI.OPSREC.LISTING(COMDHDR) PRINT HEADER
//         DD  DISP=(OLD,PASS),DSN=&&HDRDT                DATE ON HDR
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
//*
//GENER2   EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=(OLD,PASS),DSN=&&BYBKP,RECFM=FBA      BY BKP SUITE
//         DD  DISP=(OLD,PASS),DSN=&&CARTS,RECFM=FBA      ALL CARTS
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
//*
//GENER3   EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=SHR,DSN=BQIBI.OPSREC.LISTING(DISK2)   DISK MAP
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
//*
//GENER4   EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=(OLD,PASS),DSN=&&SYS1,RECFM=FBA       MVSSYS1 DASD
//         DD  DISP=(OLD,PASS),DSN=&&SYS3,RECFM=FBA       MVSSYS3 DASD
//         DD  DISP=(OLD,PASS),DSN=&&SYS2,RECFM=FBA       MVSSYS2 DASD
//         DD  DISP=(OLD,PASS),DSN=&&DASD,RECFM=FBA       ALL DASD
//         DD  DISP=(OLD,PASS),DSN=&&SUITES,RECFM=FBA     VOLS IN SUITE
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
//*
//GENER5   EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=SHR,DSN=BQIBI.OPSREC.LISTING(XERLAND) SET L'SCAPE
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
//*
//*   AN E-MAIL STEP CAN BE ADDED HERE TO SEND THE REPORTS TO THE
//*   BACK-UP SITE, SO THAT THE CORRECT DASD PROFILE CAN AUTOMATICALLY
//*   BE GENERATED BY THEM.
```

```
//*
```

SOPBK176

```
)CM ****************************************************************
)CM * +++ CREATE AN IPLABLE STAND-ALONE DFDSS CART FOR USE AT    +++ *
)CM * +++ RECOVERY. REQUIRES AN 'NL' CARTRIDGE.                  +++ *
)CM ****************************************************************
//&BKJBNM JOB ,'CREATE S/ALONE CART',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=A
//*
//BUILDSA EXEC PGM=ADRDSSU,PARM='UTILMSG=YES',REGION=4M
//*
//SAMODS   DD  DSN=SYS1.SADRYLIB,DISP=SHR
//TAPEDD   DD  DSN=STNDALON.CART,UNIT=CARTMAN,LABEL=(,NL),
//             DCB=(DSORG=PS,RECFM=U,BLKSIZE=32760,LRECL=32760),
//             VOL=SER=SA0001,DISP=(NEW,PASS)
//*
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
  BUILDSA -
  INDD(SAMODS) -
  OUTDD(TAPEDD) -
  IPL(TAPE)
/*
```

SOPBK177

```
)CM ****************************************************************
)CM * +++ DUMMY JOB (BR14) TO BE SUBMITTED AT BKUPSITE FOR       +++ *
)CM * +++ VOLUMES THAT ARE EXCLUDED FROM RESTORE (THIS GIVES     +++ *
)CM * +++ THE OPS SOMETHING TO TICK OFF THEIR LIST OF DISKS).    +++ *
)CM * +++ THIS IS NECESSARY AS THEY WILL SELECT "ALL" TO         +++ *
)CM * +++ RESTORE ALL DISKS AND THIS IS A NEAT WAY OF AVOIDING   +++ *
)CM * +++ JCL ERRORS, ETC.                                      +++ *
)CM ****************************************************************
//&BKJBNM JOB ,'&DISVOL (NO RESTORE)',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=S
//* DUMMY JOB SUBMITTED FOR "&DISVOL", AS THIS DOES NOT NORMALLY
//* GET RESTORED AT BKUPSITE...
//BR14    EXEC PGM=IEFBR14
//SYSPRINT DD  SYSOUT=*
```

SOPBK178

```
)CM ****************************************************************
)CM * +++ JOB TO EJECT CARTS FROM THE ACS. USES AS INPUT THE      +++ *
)CM * +++ COMMAND MEMBER IN 'BQIBI.OPSREC.EJECT' THAT HAS BEEN   +++ *
)CM * +++ CREATED USING OPTION '7' ON THE BACKUP PANELS.         +++ *
```

```
)CM * +++                                                          +++ *
)CM * +++ THE INPUT MEMBER IS DELETED FOLLOWING THE RUN, SO IT   +++ *
)CM * +++ MUST BE RECREATED IF YOU NEED TO RERUN THIS JOB.       +++ *
)CM ***************************************************************
//&BKJBNM JOB ,'EJECT WEEKLY CARTS',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=L
//STEP1    EXEC PGM=SLUADMIN
//STEPLIB  DD  DSN=SYS4.PSLS.SLSLINK,DISP=SHR
//SLSCNTL  DD  DSN=SYS4.PSLS.DBASEPRM,DISP=SHR
//SLSPRINT DD  SYSOUT=*
//SORTWKØ1 DD  UNIT=SYSDA,SPACE=(CYL,19)
//SYSOUT   DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SLSIN    DD  DISP=SHR,DSN=BQIBI.OPSREC.EJECT(ALL)
//*
//         IF  RC LT 5 THEN
//DELMEMB  EXEC PGM=IKJEFTØ1,DYNAMNBR=5Ø
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DUMMY
//SYSTSIN  DD  *
 DELETE 'BQIBI.OPSREC.EJECT(ALL)'
/*
//        ELSE
//OPSWARN EXEC PGM=IKJEFTØ1,PARM='%EJMSG &USRID &BKJBNM'
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSEXEC  DD  DISP=SHR,DSN=BQIBI.OPSREC.REXX
//        ENDIF
//
```

SOPBK179

```
)CM ***************************************************************
)CM * +++ GENERATE JUST THE CART LISTINGS FOR A SELECTED SUITE   +++ *
)CM * +++ AND THEN ISSUE THE COMMANDS TO EJECT THE CARTS FROM    +++ *
)CM * +++ THE ACS.                                               +++ *
)CM * +++                                                       +++ *
)CM * +++ THIS IS USED WHEN CARTS MUST BE EJECTED DURING THE     +++ *
)CM * +++ WEEK (EG THE NIGHTLY EXTRA BACK-UPS).                 +++ *
)CM ***************************************************************
//&BKJBNM JOB ,'CART EJECT: &CMPRM',NOTIFY=&USRID,
//    MSGLEVEL=(1,1),MSGCLASS=E,CLASS=A
//*--------------- GET LIST OF SYS8 GDGS -------------------------
//LISTC1  EXEC PGM=IDCAMS
//SYSPRINT DD  DISP=(,PASS),SPACE=(CYL,(1Ø,1Ø)),DSN=&&LC1,UNIT=SYSDA
//SYSIN    DD  *
 LISTC LVL(SYS8) GDG
/*
```

```
//*---------- EXTRACT OUT THE SELECTED GDG BASE NAMES ------------
//RUNREXX1 EXEC PGM=IKJEFT01,DYNAMNBR=200,
//           PARM=('%GENLIST1 &CMPRM')
//SYSTSIN  DD  DUMMY
//LISTING  DD  DISP=(OLD,PASS),DSN=&&LC1
//OUTPUT   DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&NAMES,UNIT=SYSDA
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DSN=BQIBI.OPSREC.REXX,DISP=SHR
//SYSPROC  DD  DSN=SYS3.CLIB,DISP=SHR
//*---------------- LIST THE SELECTED ENTRY -----------------------
//LISTC2   EXEC PGM=IDCAMS
//SYSPRINT DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&LC2,UNIT=SYSDA
//SYSIN    DD  DISP=(OLD,PASS),DSN=&&NAMES
//*---------------- PRODUCE 2 MORE LISTINGS -----------------------
//RUNREXX2 EXEC PGM=IKJEFT01,DYNAMNBR=200,
//           PARM=('%GENLIST2')
//SYSTSIN  DD  DUMMY
//LISTING  DD  DISP=(OLD,PASS),DSN=&&LC2
//OUTPUT   DD  DUMMY             NOT NEEDED HERE (SEE SOPBK175)
//VOLIDS   DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&VOLID,UNIT=SYSDA
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DSN=BQIBI.OPSREC.REXX,DISP=SHR
//*---------------- SORT VOLIDS INTO ORDER ------------------------
//SORT1    EXEC PGM=SORT,REGION=2048K
//SORTIN   DD  DISP=(OLD,PASS),DSN=&&VOLID
//SORTOUT  DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&VOLSRT,UNIT=SYSDA
//SORTLIB  DD  DISP=SHR,DSNAME=SYS1.SORTLIB
//SORTWK01 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWK02 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SORTWK03 DD  SPACE=(CYL,25,,CONTIG),UNIT=SYSDA
//SYSOUT   DD  SYSOUT=*
//SYSIN    DD  *
  SORT FIELDS=(5,6,CH,A)
/*
//*---------------- PRODUCE LIST OF THE CARTS ---------------------
//RUNREXX3 EXEC PGM=IKJEFT01,DYNAMNBR=200,
//           PARM=('%GENLIST3 &CMPRM &CMCAP')
//SYSTSIN  DD  DUMMY
//VOLIDS   DD  DISP=(OLD,PASS),DSN=&&VOLSRT
//OUTPUT   DD  DISP=(,PASS),SPACE=(CYL,(5,5)),DSN=&&CARTS,UNIT=SYSDA
//EJECTS   DD  DISP=(,PASS),SPACE=(CYL,(1,1)),DSN=&&EJECT,UNIT=SYSDA,
//         LRECL=80,RECFM=FB,BLKSIZE=800
//SYSTSPRT DD  SYSOUT=*
//SYSEXEC  DD  DSN=BQIBI.OPSREC.REXX,DISP=SHR
//*
//GENER1A  EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=SHR,DSN=BQIBI.OPSREC.LISTING(XERPORT) SET PORTRAIT
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
```

```
//*
//GENER2A  EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=(OLD,PASS),DSN=&&CARTS,RECFM=FBA  JUST CART NO'S
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
//*
//GENER3A  EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DISP=SHR,DSN=BQIBI.OPSREC.LISTING(XERLAND) SET L'SCAPE
//SYSUT2   DD  SYSOUT=A,DEST=OP1
//SYSIN    DD  DUMMY
//*
//EJECT    EXEC PGM=SLUADMIN
//STEPLIB  DD  DSN=SYS4.PSLS.SLSLINK,DISP=SHR
//SLSCNTL  DD  DSN=SYS4.PSLS.DBASEPRM,DISP=SHR
//SLSPRINT DD  SYSOUT=*
//SORTWKØ1 DD  UNIT=SYSDA,SPACE=(CYL,19)
//SYSOUT   DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SLSIN    DD  DISP=(OLD,PASS),DSN=&&EJECT
//*
```

ISPF PANELS

POPBK171

```
)ATTR
      ¬ TYPE(OUTPUT) INTENS(HIGH) CAPS(OFF)
      £ TYPE(OUTPUT) INTENS(LOW)  CAPS(OFF)
      $ TYPE(TEXT)   INTENS(HIGH) CAPS(OFF)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Processing on:¬sys %@-@
%@ @¬ATCOMD      %@ @
+
+
+                  Select one of the following  ==>_opsel+
+
+            -------------- Onsite Back-ups and Restores -------------
%             1) +Submit$Backup+Jobs
%             2) +Submit$Restore+Jobs%at "your site name"
%             3) +SDSF Display of Backup/Restore Jobs
%             4) +Display Backup/Restore Suite$Contents+
%             5) +Create a Standalone DFDSS IPLable cart (on request)
+            -------------- Weekly Onsite Processing -----------------
%             6) +Generate%"backup site"+Restores & Backup 'ONEPAC'
%             7) %"backup site"+Print (only after 6 has finished)
%             8) +Eject%"backup site"+Carts (after 7 has finished)
```

```
%              9) +Selective Eject of backup carts        (as required)
+            ------------------ Offsite RECOVERY ---------------------
%              R) +Submit%Restore+jobs%at "backup site"+
+
+
%  PF3+to End.
)PROC
  VER (&OPSEL,NB,LIST,1,2,3,4,5,6,7,8,9,R)
)END
```

POPBK172

```
)ATTR
  $ TYPE(INPUT)  INTENS(HIGH) PAD(_)
  ¬ TYPE(OUTPUT) INTENS(HIGH)
  # AREA(SCRL) EXTEND(ON)
)BODY WIDTH(8Ø) EXPAND(@@)
+@-@ Dump/Restore -%Dump+Volume(s) for:¬suite  %@-@
%@ @¬ATCOMD      %@ @
+
%  PF7+- Up,%PF8+-Down,%PF3+-End.               Jobname Prefix: ¬jobnm
+
+  Enter%"ALL"+to submit Full Backups,          Volumes in
suite:¬volcnt+
+  or the%number+for a single one: ===>$BKU+
#SCRLAREA                                                                 #
#                                                                         #
#                                                                         #
#                                                                         #
#                                                                         #
#                                                                         #
#                                                                         #
#                                                                         #
+
+ Press%ENTER+to process input,%PF3+to End.
)AREA SCRLAREA
   +
=========================================================================
+|    Volid    |    Volid    |    Volid    |    Volid    |    Volid    |
+|-------------------------------------------------------------------------|
+|¬B1  ¬BKVL1 +|¬B2  ¬BKVL2 +|¬B3  ¬BKVL3 +|¬B4  ¬BKVL4 +|¬B5  ¬BKVL5 +|
+|-------------------------------------------------------------------------|
+|¬B6  ¬BKVL6 +|¬B7  ¬BKVL7 +|¬B8  ¬BKVL8 +|¬B9  ¬BKVL9 +|¬B1Ø ¬BKVL1Ø+|
+|-------------------------------------------------------------------------|
+|¬B11 ¬BKVL11+|¬B12 ¬BKVL12+|¬B13 ¬BKVL13+|¬B14 ¬BKVL14+|¬B15 ¬BKVL15+|
+|-------------------------------------------------------------------------|
+|¬B16 ¬BKVL16+|¬B17 ¬BKVL17+|¬B18 ¬BKVL18+|¬B19 ¬BKVL19+|¬B2Ø ¬BKVL2Ø+|
+|-------------------------------------------------------------------------|
+|¬B21 ¬BKVL21+|¬B22 ¬BKVL22+|¬B23 ¬BKVL23+|¬B24 ¬BKVL24+|¬B25 ¬BKVL25+|
```

```
+|-----------------------------------------------------------------|
+|¬B26 ¬BKVL26+|¬B27 ¬BKVL27+|¬B28 ¬BKVL28+|¬B29 ¬BKVL29+|¬B3Ø ¬BKVL3Ø+|
+|-----------------------------------------------------------------|
+|¬B31 ¬BKVL31+|¬B32 ¬BKVL32+|¬B33 ¬BKVL33+|¬B34 ¬BKVL34+|¬B35 ¬BKVL35+|
+|-----------------------------------------------------------------|
+|¬B36 ¬BKVL36+|¬B37 ¬BKVL37+|¬B38 ¬BKVL38+|¬B39 ¬BKVL39+|¬B4Ø ¬BKVL4Ø+|
+|-----------------------------------------------------------------|
+|¬B41 ¬BKVL41+|¬B42 ¬BKVL42+|¬B43 ¬BKVL43+|¬B44 ¬BKVL44+|¬B45 ¬BKVL45+|
+|-----------------------------------------------------------------|
+|¬B46 ¬BKVL46+|¬B47 ¬BKVL47+|¬B48 ¬BKVL48+|¬B49 ¬BKVL49+|¬B5Ø ¬BKVL5Ø+|
+|-----------------------------------------------------------------|
+|¬B51 ¬BKVL51+|¬B52 ¬BKVL52+|¬B53 ¬BKVL53+|¬B54 ¬BKVL54+|¬B55 ¬BKVL55+|
+|-----------------------------------------------------------------|
+|¬B56 ¬BKVL56+|¬B57 ¬BKVL57+|¬B58 ¬BKVL58+|¬B59 ¬BKVL59+|¬B6Ø ¬BKVL6Ø+|
+ ================================================================
)INIT
  IF (&BKVL1 = '*NONE*')
    .ATTR (BKVL1) = 'INTENS(LOW)'
    .ATTR (B1)    = 'INTENS(LOW)'
  IF (&BKVL2 = '*NONE*')
    .ATTR (BKVL2) = 'INTENS(LOW)'
    .ATTR (B2)    = 'INTENS(LOW)'
and so on until.............
  IF (&BKVL59 = '*NONE*')
    .ATTR (BKVL59) = 'INTENS(LOW)'
    .ATTR (B59)    = 'INTENS(LOW)'
  IF (&BKVL6Ø = '*NONE*')
    .ATTR (BKVL6Ø) = 'INTENS(LOW)'
    .ATTR (B6Ø)    = 'INTENS(LOW)'
)PROC
  VER (&BKU,NB,LIST,ALL,STATUS,1,2,3,4,5,6,7,8,9,1Ø,11,12,13,14,15,
      16,17,18,19,2Ø,21,22,23,24,25,26,27,28,29,3Ø,31,32,33,34,35,
      36,37,38,39,4Ø,41,42,43,44,45,46,47,48,49,5Ø,51,52,53,54,55,
      56,57,58,59,6Ø)
  IF (&BKU = 1)
    IF (&BKVL1 = '*NONE*')
      .MSG = MBKP17ØD
  IF (&BKU = 2)
    IF (&BKVL2 = '*NONE*')
      .MSG = MBKP17ØD
and so on until.............
  IF (&BKU = 59)
    IF (&BKVL59 = '*NONE*')
      .MSG = MBKP17ØD
  IF (&BKU = 6Ø)
    IF (&BKVL6Ø = '*NONE*')
      .MSG = MBKP17ØD
)END
```

POPBK174

```
)ATTR DEFAULT(%+$)
      ¬ TYPE(OUTPUT) INTENS(LOW)
      £ TYPE(TEXT)   INTENS(LOW) HILITE(BLINK)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Dump Volume(s)%@-@
%@ @¬ATCOMD      %@ @
+
%@ @ ------------------------------------------------------- @ @
%@ @|                                                       |@ @
%@ @| The Back-up Suite you have selected (¬suite  %) is£not% |@ @
%@ @|            supposed to run on this system!!!          |@ @
%@ @|                                                       |@ @
%@ @|       ¬bkms1                                    %     |@ @
%@ @|                                                       |@ @
%@ @ ------------------------------------------------------- @ @
+
)PROC
)END
```

POPBK175

```
)ATTR DEFAULT(%+$)
/* !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
/* WARNING - hex attribute characters follow...use 'HEX ON' to view
/* There is an attribute character for each possible output field.
/* For the submission of this article I have replaced the hex values
/* with '?'. This panel needs to have these altered to the actual
/* hex values x'Ø1' through to x'24' for the 24 groups of fields.
/* !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      £ TYPE(OUTPUT) INTENS(HIGH)
      # TYPE(INPUT)  INTENS(LOW) PAD(_)
      ? TYPE(OUTPUT) INTENS(LOW)           this is actually x'Ø1'
      ? TYPE(OUTPUT) INTENS(LOW)           this is actually x'Ø2'
and so on until..............
      ? TYPE(OUTPUT) INTENS(LOW)           this is actually x'23'
      ? TYPE(OUTPUT) INTENS(LOW)           this is actually x'24'
/* !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
/* BEWARE!!! This panel uses hex attribute characters.
/* !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Select Suite for£sltp    %on:£sys %@-@
%@ @£ATCOMD      %@ @
+
+
+
+    Select required suite%suffix+from the following and press%ENTER:#Z+
+    Standard backups for£sys2+are%highlighted.+
+ Replace the following '?' with
```

```
+ x'Ø1' x'Ø1'       x'Ø2' x'Ø2'       x'Ø3' x'Ø3'       x'Ø4' x'Ø4' etc
+   ?BK1 ?SUITE1  +   ?BK2 ?SUITE2  +   ?BK3 ?SUITE3  +   ?BK4 ?SUITE4 +
+   ?BK5 ?SUITE5  +   ?BK6 ?SUITE6  +   ?BK7 ?SUITE7  +   ?BK8 ?SUITE8 +
+   ?BK9 ?SUITE9  +   ?BK1Ø?SUITE1Ø +   ?BK11?SUITE11 +   ?BK12?SUITE12+
+   ?BK13?SUITE13 +   ?BK14?SUITE14 +   ?BK15?SUITE15 +   ?BK16?SUITE16+
+   ?BK17?SUITE17 +   ?BK18?SUITE18 +   ?BK19?SUITE19 +   ?BK2Ø?SUITE2Ø+
+   ?BK21?SUITE21 +   ?BK22?SUITE22 +   ?BK23?SUITE23 +   ?BK24?SUITE24+
+   ?BK25?SUITE25 +   ?BK26?SUITE26 +   ?BK27?SUITE27 +   ?BK28?SUITE28+
+   ?BK29?SUITE29 +   ?BK3Ø?SUITE3Ø +   ?BK31?SUITE31 +   ?BK32?SUITE32+
+   ?BK33?SUITE33 +   ?BK34?SUITE34 +   ?BK35?SUITE35 +   ?BK36?SUITE36+
+                                                                      +
+  Press%PF3+to End
)INIT
.ZVARS ='(ASUITE)'
&SYS2 = &SYS
/*
/* WARNING - hex attribute characters follow...use 'HEX ON' to view
/*
IF (&BK1ATTR = 'HI')
    .ATTRCHAR(?) = 'INTENS(HIGH)'   actually x'Ø1'
IF (&BK2ATTR = 'HI')
    .ATTRCHAR(?) = 'INTENS(HIGH)'   actually x'Ø2'
and so on until..............
IF (&BK35ATTR = 'HI')
    .ATTRCHAR(?) = 'INTENS(HIGH)'   actually x'23'
IF (&BK36ATTR = 'HI')
    .ATTRCHAR(?) = 'INTENS(HIGH)'   actually x'24'
)PROC
VER(&ASUITE,NB,LIST,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,
 X,Y,Z,Ø,1,2,3,4,5,6,7,8,9)
)END
```

POPBK176

```
)ATTR
      ¬ TYPE(OUTPUT) INTENS(HIGH)
      $ TYPE(TEXT)   INTENS(HIGH) HILITE(BLINK)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Processing on:¬sys %@-@
%@ @¬ATCOMD      %@ @
+
%@ @ ------------------------------------------------------------ @ @
%@ @| !The DATASET containing the¬bkmiss       %cannot be found! |@ @
%@ @|               Please contact Systems$IMMEDIATELY.%         |@ @
%@ @ ------------------------------------------------------------ @ @
+
+  Press%ENTER+or%PF3+to Continue.
)INIT
)PROC
)END
```

POPBK177

```
)ATTR
  $ TYPE(INPUT)  INTENS(HIGH) PAD(_)
  ¬ TYPE(OUTPUT) INTENS(HIGH)
  # AREA(SCRL) EXTEND(ON)
)BODY WIDTH(8Ø) EXPAND(@@)
+@-@ Dump/Restore -%Restore+Volume for:¬suite  %@-@
%@ @¬ATCOMD      %@ @
+
%  PF7+- Up,%PF8+- Down,%PF3+- End.                Jobname Prefix:
¬jobnm  +
+  Select the number to be restored:    ===>$Z +    Volumes in
suite:¬volcnt+
+  Enter generation number (eg Ø, -1):  ===>$Z +
+  Target address: $BTGT+ Target volid: ===>$BTGTVL+
#SCRLAREA                                                                #
#                                                                        #
#                                                                        #
#                                                                        #
#                                                                        #
#                                                                        #
#                                                                        #
#                                                                        #
+
+ Press%ENTER+to process input,%PF3+to End.
)AREA SCRLAREA
   +
=========================================================================
+|    Volid    |    Volid    |    Volid    |    Volid    |    Volid    |
+|-------------------------------------------------------------------------|
+|¬B1  ¬BKVL1 +|¬B2  ¬BKVL2 +|¬B3  ¬BKVL3 +|¬B4  ¬BKVL4 +|¬B5  ¬BKVL5 +|
+|-------------------------------------------------------------------------|
+|¬B6  ¬BKVL6 +|¬B7  ¬BKVL7 +|¬B8  ¬BKVL8 +|¬B9  ¬BKVL9 +|¬B1Ø ¬BKVL1Ø+|
+|-------------------------------------------------------------------------|
+|¬B11 ¬BKVL11+|¬B12 ¬BKVL12+|¬B13 ¬BKVL13+|¬B14 ¬BKVL14+|¬B15 ¬BKVL15+|
+|-------------------------------------------------------------------------|
+|¬B16 ¬BKVL16+|¬B17 ¬BKVL17+|¬B18 ¬BKVL18+|¬B19 ¬BKVL19+|¬B2Ø ¬BKVL2Ø+|
+|-------------------------------------------------------------------------|
+|¬B21 ¬BKVL21+|¬B22 ¬BKVL22+|¬B23 ¬BKVL23+|¬B24 ¬BKVL24+|¬B25 ¬BKVL25+|
+|-------------------------------------------------------------------------|
+|¬B26 ¬BKVL26+|¬B27 ¬BKVL27+|¬B28 ¬BKVL28+|¬B29 ¬BKVL29+|¬B3Ø ¬BKVL3Ø+|
+|-------------------------------------------------------------------------|
+|¬B31 ¬BKVL31+|¬B32 ¬BKVL32+|¬B33 ¬BKVL33+|¬B34 ¬BKVL34+|¬B35 ¬BKVL35+|
+|-------------------------------------------------------------------------|
+|¬B36 ¬BKVL36+|¬B37 ¬BKVL37+|¬B38 ¬BKVL38+|¬B39 ¬BKVL39+|¬B4Ø ¬BKVL4Ø+|
+|-------------------------------------------------------------------------|
+|¬B41 ¬BKVL41+|¬B42 ¬BKVL42+|¬B43 ¬BKVL43+|¬B44 ¬BKVL44+|¬B45 ¬BKVL45+|
+|-------------------------------------------------------------------------|
+|¬B46 ¬BKVL46+|¬B47 ¬BKVL47+|¬B48 ¬BKVL48+|¬B49 ¬BKVL49+|¬B5Ø ¬BKVL5Ø+|
```

```
+|-----------------------------------------------------------------|
+|¬B51 ¬BKVL51+|¬B52 ¬BKVL52+|¬B53 ¬BKVL53+|¬B54 ¬BKVL54+|¬B55 ¬BKVL55+|
+|-----------------------------------------------------------------|
+|¬B56 ¬BKVL56+|¬B57 ¬BKVL57+|¬B58 ¬BKVL58+|¬B59 ¬BKVL59+|¬B6Ø ¬BKVL6Ø+|
   +
=================================================================
)INIT
  .ZVARS = '(BKU BGN)'
  IF (&DOSUB ¬= 'N')
    &BTGT = ''
    &BTGTVL = ''
    &BKU = ''
    &BGN = ''
  IF (&BKVL1 = '*NONE*')
    .ATTR (BKVL1) = 'INTENS(LOW)'
    .ATTR (B1)    = 'INTENS(LOW)'
  IF (&BKVL2 = '*NONE*')
    .ATTR (BKVL2) = 'INTENS(LOW)'
    .ATTR (B2)    = 'INTENS(LOW)'
and so on until.............
  IF (&BKVL59 = '*NONE*')
    .ATTR (BKVL59) = 'INTENS(LOW)'
    .ATTR (B59)    = 'INTENS(LOW)'
  IF (&BKVL6Ø = '*NONE*')
    .ATTR (BKVL6Ø) = 'INTENS(LOW)'
    .ATTR (B6Ø)    = 'INTENS(LOW)'
)PROC
  VER (&BKU,NB,LIST,1,2,3,4,5,6,7,8,9,1Ø,11,12,13,14,15,16,17,
      18,19,2Ø,21,22,23,24,25,26,27,28,29,3Ø,31,32,33,34,35,36,
      37,38,39,4Ø,41,42,43,44,45,46,47,48,49,5Ø,51,52,53,54,55,
      56,57,58,59,6Ø)
  VER (&BGN,NB,LIST,Ø,-1,-2,-3,-4,-5)
  VER (&BTGT,NB,HEX)
  VER (&BTGT,LEN,GE,3)
  VER (&BTGTVL,NB)
  VER (&BTGTVL,LEN,EQ,6)
  /* ------------------------------------------ */
  /*  Ensure they haven't selected an empty entry  */
  /* ------------------------------------------ */
  IF (&BKU = 1)
    IF (&BKVL1 = '*NONE*')
      .MSG = MBKP17ØD
  IF (&BKU = 2)
    IF (&BKVL2 = '*NONE*')
      .MSG = MBKP17ØD
and so on until.............
  IF (&BKU = 59)
    IF (&BKVL59 = '*NONE*')
      .MSG = MBKP17ØD
  IF (&BKU = 6Ø)
```

```
    IF (&BKVL6Ø = '*NONE*')
      .MSG = MBKP17ØD
)END
```

POPBK178

```
)ATTR DEFAULT(%+_)
      ! TYPE(OUTPUT) INTENS(LOW) CAPS(OFF)
      ¬ TYPE(OUTPUT) INTENS(HIGH)
      # AREA(SCRL) EXTEND(ON)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Display Backup Suite Contents @-@
%@ @¬ATCOMD      %@ @
+
%                PF7+-Up                          Suite:   ¬suite  +
%                PF8+-Down                        Jobname: ¬bkjbnm +
%                PF3+-End                         Volumes: ¬volcnt+
#SCRLAREA                                                          #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
+
+Press%PF3+to%End
)AREA SCRLAREA
+  ==Volser==Addr===Devt=== ==Volser==Addr===Devt===
==Volser==Addr===Devt===
+ | ¬BKVL1  ¬BKAD1 ¬BKDV1 +| ¬BKVL2  ¬BKAD2 ¬BKDV2 +| ¬BKVL3  ¬BKAD3
¬BKDV3 +|
+ | ¬BKVL4  ¬BKAD4 ¬BKDV4 +| ¬BKVL5  ¬BKAD5 ¬BKDV5 +| ¬BKVL6  ¬BKAD6
¬BKDV6 +|
+ | ¬BKVL7  ¬BKAD7 ¬BKDV7 +| ¬BKVL8  ¬BKAD8 ¬BKDV8 +| ¬BKVL9  ¬BKAD9
¬BKDV9 +|
+ | ¬BKVL1Ø ¬BKAD1Ø¬BKDV1Ø+| ¬BKVL11 ¬BKAD11¬BKDV11+| ¬BKVL12
¬BKAD12¬BKDV12+|
+ | ¬BKVL13 ¬BKAD13¬BKDV13+| ¬BKVL14 ¬BKAD14¬BKDV14+| ¬BKVL15
¬BKAD15¬BKDV15+|
+ | ¬BKVL16 ¬BKAD16¬BKDV16+| ¬BKVL17 ¬BKAD17¬BKDV17+| ¬BKVL18
¬BKAD18¬BKDV18+|
+ | ¬BKVL19 ¬BKAD19¬BKDV19+| ¬BKVL2Ø ¬BKAD2Ø¬BKDV2Ø+| ¬BKVL21
¬BKAD21¬BKDV21+|
+ | ¬BKVL22 ¬BKAD22¬BKDV22+| ¬BKVL23 ¬BKAD23¬BKDV23+| ¬BKVL24
¬BKAD24¬BKDV24+|
+ | ¬BKVL25 ¬BKAD25¬BKDV25+| ¬BKVL26 ¬BKAD26¬BKDV26+| ¬BKVL27
¬BKAD27¬BKDV27+|
+ | ¬BKVL28 ¬BKAD28¬BKDV28+| ¬BKVL29 ¬BKAD29¬BKDV29+| ¬BKVL3Ø
¬BKAD3Ø¬BKDV3Ø+|
+ | ¬BKVL31 ¬BKAD31¬BKDV31+| ¬BKVL32 ¬BKAD32¬BKDV32+| ¬BKVL33
```

```
¬BKAD33¬BKDV33+|
+ | ¬BKVL34 ¬BKAD34¬BKDV34+| ¬BKVL35 ¬BKAD35¬BKDV35+| ¬BKVL36
¬BKAD36¬BKDV36+|
+ | ¬BKVL37 ¬BKAD37¬BKDV37+| ¬BKVL38 ¬BKAD38¬BKDV38+| ¬BKVL39
¬BKAD39¬BKDV39+|
+ | ¬BKVL4Ø ¬BKAD4Ø¬BKDV4Ø+| ¬BKVL41 ¬BKAD41¬BKDV41+| ¬BKVL42
¬BKAD42¬BKDV42+|
+ | ¬BKVL43 ¬BKAD43¬BKDV43+| ¬BKVL44 ¬BKAD44¬BKDV44+| ¬BKVL45
¬BKAD45¬BKDV45+|
+ | ¬BKVL46 ¬BKAD46¬BKDV46+| ¬BKVL47 ¬BKAD47¬BKDV47+| ¬BKVL48
¬BKAD48¬BKDV48+|
+ | ¬BKVL49 ¬BKAD49¬BKDV49+| ¬BKVL5Ø ¬BKAD5Ø¬BKDV5Ø+| ¬BKVL51
¬BKAD51¬BKDV51+|
+ | ¬BKVL52 ¬BKAD52¬BKDV52+| ¬BKVL53 ¬BKAD53¬BKDV53+| ¬BKVL54
¬BKAD54¬BKDV54+|
+ | ¬BKVL55 ¬BKAD55¬BKDV55+| ¬BKVL56 ¬BKAD56¬BKDV56+| ¬BKVL57
¬BKAD57¬BKDV57+|
+ | ¬BKVL58 ¬BKAD58¬BKDV58+| ¬BKVL59 ¬BKAD59¬BKDV59+| ¬BKVL6Ø
¬BKAD6Ø¬BKDV6Ø+|
+
================================================================================
)INIT
  &PRTFLG = 'NO'
  IF (&BKVL1 = '*NONE*')
    .ATTR (BKVL1) = 'INTENS(LOW)'
  IF (&BKVL2 = '*NONE*')
    .ATTR (BKVL2) = 'INTENS(LOW)'
and so on until..............
  IF (&BKVL58 = '*NONE*')
    .ATTR (BKVL58) = 'INTENS(LOW)'
  IF (&BKVL59 = '*NONE*')
    .ATTR (BKVL59) = 'INTENS(LOW)'
  IF (&BKVL6Ø = '*NONE*')
    .ATTR (BKVL6Ø) = 'INTENS(LOW)'
)PROC
)END
```

POPBK17A

```
)ATTR DEFAULT(%+$)
      £ TYPE(OUTPUT) INTENS(HIGH)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Restore Volume%@-@
%@ @£ATCOMD      %@ @
+
+@ @\-----------------------------------------------------------/@ @
+@ @|                                                           |@ @
+@ @| You have selected a%RESTORE+with the following details: |@ @
+@ @|                                                           |@ @
+@ @|        Back-up generation                                 |@ @
```

```
+@ @|          to Restore.........£bgn+                          |@ @
+@ @|        Source Volid.........£volid +                       |@ @
+@ @|        Target Volid.........£btgtvl+                       |@ @
+@ @|        Target Address.......£btgt+                         |@ @
+@ @|                                                            |@ @
+@ @|                                                            |@ @
+@ @|      Press%ENTER+to Continue, or%PF3+to Cancel Now         |@ @
+@ @|                                                            |@ @
+@ @/------------------------------------------------------------\@ @
+
)PROC
)END
```

POPBK17B

```
)ATTR DEFAULT(%+$)
      ¬ TYPE(OUTPUT) INTENS(HIGH)
      £ TYPE(TEXT)   INTENS(HIGH) HILITE(BLINK)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Processing on:¬sys %@-@
%@ @¬ATCOMD      %@ @
+
%@ @\-----------------------------------------------------------------/@ @
%@ @|                                                                 |@ @
%@ @|  The 'BKUPSITE Restore Selection Record' for this system        |@ @
%@ @| cannot be located..... Please contact Systems£IMMEDIATELY%      |@ @
%@ @|                                                                 |@ @
%@ @/-----------------------------------------------------------------\@ @
+
)PROC
)END
```

POPBK17C

```
)ATTR DEFAULT(%+$)
      ¬ TYPE(OUTPUT) INTENS(HIGH)
      £ TYPE(TEXT)   INTENS(HIGH) HILITE(BLINK)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Dump/Restore - Processing on:¬sys %@-@
%@ @¬ATCOMD      %@ @
+
%@ @\-----------------------------------------------------------------/@ @
%@ @|                                                                 |@ @
%@ @| The number of entries in the 'System Identification Record'     |@ @
%@ @| does not equal the number found in the 'BKUPSITE Restore        |@ @
%@ @| Selection Record'..... Please contact Systems£IMMEDIATELY%      |@ @
%@ @|                                                                 |@ @
%@ @/-----------------------------------------------------------------\@ @
+
```

```
)PROC
)END
```

POPBK17D

```
)ATTR DEFAULT(%+$)
      # TYPE(INPUT)  INTENS(LOW) PAD(_)
      ¬ TYPE(OUTPUT) INTENS(HIGH)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Recovery at BKUPSITE%@-@
%@ @¬ATCOMD      %@ @
+
%@ @ ------------------------------------------------------- @ @
%@ @|                                                       |@ @
%@ @|   Please enter the system id that you want to perform  |@ @
%@ @|           recovery for from the list below:           |@ @
%@ @|                                                       |@ @
%@ @|                    SYS1   SYS2   SYS3                 |@ @
%@ @|                                                       |@ @
%@ @|                   Recovery id ==>#sys %               |@ @
%@ @|                                                       |@ @
%@ @|                                                       |@ @
%@ @ ------------------------------------------------------- @ @
+
%ENTER -+Continue
%PF3   -+End
)PROC
  VER (&SYS,NB,LIST,SYS1,SYS2,SYS3)
)END
```

POPBK17E

```
)ATTR
  $ TYPE(INPUT)  INTENS(HIGH) PAD(_)
  ¬ TYPE(OUTPUT) INTENS(HIGH)
  # AREA(SCRL) EXTEND(ON)
)BODY WIDTH(8Ø) EXPAND(@@)
+@-@ Dump/Restore -%Offsite Recovery+for:¬suite  %@-@
%@ @¬ATCOMD      %@ @
+
%  PF7+- Up,%PF8+-Down,%PF3+-End.             Jobname Prefix: ¬bkjbnm +
+  Enter%"ALL"+to submit Full Suite Restore,   Volumes in suite:¬volcnt+
+  or the%number+for a single one: ===>$BKU+   Generation (Ø,-1,-2):$Z +
#SCRLAREA                                                              #
#                                                                      #
#                                                                      #
#                                                                      #
#                                                                      #
#                                                                      #
```

```
#                                                                      #
#                                                                      #
+
+ Press%ENTER+to process input,%PF3+to End.
)AREA SCRLAREA
   +
=====================================================================
+|    Volid    |    Volid    |    Volid    |    Volid    |    Volid    |
+|-------------------------------------------------------------------|
+|¬B1  ¬BKVL1 +|¬B2  ¬BKVL2 +|¬B3  ¬BKVL3 +|¬B4  ¬BKVL4 +|¬B5  ¬BKVL5 +|
+|-------------------------------------------------------------------|
+|¬B6  ¬BKVL6 +|¬B7  ¬BKVL7 +|¬B8  ¬BKVL8 +|¬B9  ¬BKVL9 +|¬B1Ø ¬BKVL1Ø+|
+|-------------------------------------------------------------------|
+|¬B11 ¬BKVL11+|¬B12 ¬BKVL12+|¬B13 ¬BKVL13+|¬B14 ¬BKVL14+|¬B15 ¬BKVL15+|
+|-------------------------------------------------------------------|
+|¬B16 ¬BKVL16+|¬B17 ¬BKVL17+|¬B18 ¬BKVL18+|¬B19 ¬BKVL19+|¬B2Ø ¬BKVL2Ø+|
+|-------------------------------------------------------------------|
+|¬B21 ¬BKVL21+|¬B22 ¬BKVL22+|¬B23 ¬BKVL23+|¬B24 ¬BKVL24+|¬B25 ¬BKVL25+|
+|-------------------------------------------------------------------|
+|¬B26 ¬BKVL26+|¬B27 ¬BKVL27+|¬B28 ¬BKVL28+|¬B29 ¬BKVL29+|¬B3Ø ¬BKVL3Ø+|
+|-------------------------------------------------------------------|
+|¬B31 ¬BKVL31+|¬B32 ¬BKVL32+|¬B33 ¬BKVL33+|¬B34 ¬BKVL34+|¬B35 ¬BKVL35+|
+|-------------------------------------------------------------------|
+|¬B36 ¬BKVL36+|¬B37 ¬BKVL37+|¬B38 ¬BKVL38+|¬B39 ¬BKVL39+|¬B4Ø ¬BKVL4Ø+|
+|-------------------------------------------------------------------|
+|¬B41 ¬BKVL41+|¬B42 ¬BKVL42+|¬B43 ¬BKVL43+|¬B44 ¬BKVL44+|¬B45 ¬BKVL45+|
+|-------------------------------------------------------------------|
+|¬B46 ¬BKVL46+|¬B47 ¬BKVL47+|¬B48 ¬BKVL48+|¬B49 ¬BKVL49+|¬B5Ø ¬BKVL5Ø+|
+|-------------------------------------------------------------------|
+|¬B51 ¬BKVL51+|¬B52 ¬BKVL52+|¬B53 ¬BKVL53+|¬B54 ¬BKVL54+|¬B55 ¬BKVL55+|
+|-------------------------------------------------------------------|
+|¬B56 ¬BKVL56+|¬B57 ¬BKVL57+|¬B58 ¬BKVL58+|¬B59 ¬BKVL59+|¬B6Ø ¬BKVL6Ø+|
   +
=====================================================================
)INIT
  .ZVARS = '(BGN)'
  IF (&BKVL1 = '*NONE*')
    .ATTR (BKVL1) = 'INTENS(LOW)'
    .ATTR (B1)    = 'INTENS(LOW)'
  IF (&BKVL2 = '*NONE*')
    .ATTR (BKVL2) = 'INTENS(LOW)'
    .ATTR (B2)    = 'INTENS(LOW)'
and so on until.............
    .ATTR (BKVL3) = 'INTENS(LOW)'
  IF (&BKVL59 = '*NONE*')
    .ATTR (BKVL59) = 'INTENS(LOW)'
    .ATTR (B59)    = 'INTENS(LOW)'
  IF (&BKVL6Ø = '*NONE*')
    .ATTR (BKVL6Ø) = 'INTENS(LOW)'
    .ATTR (B6Ø)    = 'INTENS(LOW)'
```

```
)PROC
  VER (&BKU,NB,LIST,ALL,STATUS,1,2,3,4,5,6,7,8,9,1Ø,11,12,13,14,15,
      16,17,18,19,2Ø,21,22,23,24,25,26,27,28,29,3Ø,31,32,33,34,35,
      36,37,38,39,4Ø,41,42,43,44,45,46,47,48,49,5Ø,51,52,53,54,55,
      56,57,58,59,6Ø)
  IF (&BKU = 1)
    IF (&BKVL1 = '*NONE*')
      .MSG = MBKP17ØD
  IF (&BKU = 2)
    IF (&BKVL2 = '*NONE*')
      .MSG = MBKP17ØD
and so on until..............
  IF (&BKU = 59)
    IF (&BKVL59 = '*NONE*')
      .MSG = MBKP17ØD
  IF (&BKU = 6Ø)
    IF (&BKVL6Ø = '*NONE*')
      .MSG = MBKP17ØD
  VER (&BGN,NB,LIST,Ø,-1,-2)
)END
```

POPBK17F

```
)ATTR DEFAULT(%+$)
      # TYPE(INPUT)  INTENS(LOW) PAD(_)
      ¬ TYPE(OUTPUT) INTENS(HIGH)
)BODY WIDTH(8Ø) EXPAND(@@)
%@-@ Selective Eject from the ACS @-@
%@ @¬ATCOMD      %@ @
+
+@ @ ------------------------------------------------------- @ @
+@ @|                                                       |@ @
+@ @|   This option will select a single suite's cartridges  |@ @
+@ @|   to be ejected from the ACS (eg the nightly Cargo     |@ @
+@ @|   Database backups). Enter the%Backup Suite ID+below   |@ @
+@ @|   (eg 'MVS5'). You may override the CAPid if needed.   |@ @
+@ @|           A list of the carts will print on OP1.       |@ @
+@ @|                                                       |@ @
+@ @|               Listing Option ==>#cmls+                |@ @
+@ @|                                                       |@ @
+@ @|               Select CAP Id  ==>#cmcap +              |@ @
+@ @|                                                       |@ @
+@ @ ------------------------------------------------------- @ @
+
%ENTER -+Continue
%PF3   -+End
)INIT
  &CMCAP = 'ØØØ:ØØ'
)PROC
```

```
  VER (&CMLS,NB)
  &XXX = TRUNC(&CMLS,3)
  IF (&XXX NE 'MVS')
    .MSG = MBKP170U
  VER (&CMCAP,NB,LIST,000:00,000:01,001:00,001:01,MSG=MBKP170V)
)END
```

POPBK17G

```
)ATTR DEFAULT(%+$)
      # TYPE(INPUT)  INTENS(LOW) PAD(_)
      ¬ TYPE(OUTPUT) INTENS(HIGH)
)BODY WIDTH(80) EXPAND(@@)
%@-@ Run Weekly Job:¬wklyjb  %@-@
%@ @¬ATCOMD      %@ @
+
+@ @ ------------------------------------------------------ @ @
+@ @|                                                       |@ @
+@ @|   You are about to submit a weekly job to perform:    |@ @
+@ @|                                                       |@ @
+@ @|  ¬jobdesc                                        +    |@ @
+@ @|                                                       |@ @
+@ @|                                                       |@ @
+@ @|          Press%ENTER+to Continue,%PF3+to End          |@ @
+@ @|                                                       |@ @
+@ @|                                                       |@ @
+@ @ ------------------------------------------------------ @ @
+
)PROC
)END
```

Grant Carson
Systems Programmer (UK)

z/Architecture overview

OVERVIEW

With the introduction of IBM's eServer zSeries 900 processors a new architecture has evolved. It is known as z/Architecture and it is a true 64-bit processor design. The z/Architecture is a tri-modal architecture capable of executing in 24-bit, 31-bit, or 64-bit addressing modes.

The z/Architecture definition separates the control of 64-bit addressing mode from 64-bit operations. This definition allows the software to be extended to 64-bit in an evolutionary manner. 32-bit and 64-bit operations are supported in 24, 31, and 64-bit addressing mode. This allows any program to switch between addressing modes and intermix 32-bit and 64-bit operations.

Branch generation in z/Architecture always generates a 64-bit value. In 24-bit address mode the leftmost 40 bits are forced to zero. With 31-bit addressing mode the leftmost 33 bits are forced to zero

Note: 64-bit z/Architecture was previously known as ESAME (ESA Modal Extensions).

z/Architecture has:

- 64-bit address generation
- 24-bit addressing
- 31-bit addressing
- 64-bit addressing
- 64-bit General Purpose Registers (GPRs)
- 32-bit Access Registers
- 64-bit Control Registers (CRs)
- 128-bit PSW
- 16 exabyte addressing
- 32- and 64-bit arithmetic and logical operations
- 8K PSA
- New instructions and instruction types
- New format Dynamic Address Translation structures
- New format Linkage Stack and PC Linkage Structures
- New System Trace Table entries

- New Format IDAWs (64-bit I/O)
- Capable of 75-bit addressing with data spaces.

OS/390 Version 2 Release 10 was the first operating system to run in 64-bit mode on a z900 processor, but provided very little 64-bit exploitation. z/OS 1.1 added little beyond what was included with OS/390 V2R10. z/OS 1.2 is the first release that has included significant 64-bit exploitation, primarily 64-bit virtual memory capabilities for Assembler language programmers.

As of z/OS 1.2, the address space begins at address 0 and ends at 16 exabytes. z/OS 1.2 includes one major new API that enables applications to request and manage 64-bit virtual storage. The new IAVR64 macro allows applications to obtain and release 64-bit storage, as well as to share, page fix, page free, and discard memory objects.

I have written this article to provide the reader with a quick reference guide to z/Architecture. For detailed information, please refer to the *z/Architecture Principles of Operation.*

64-BIT VIRTUAL ADDRESS SPACE

In the 64-bit address space, a virtual line marks the 16-megabyte address just as the virtual line marks the 31-bit address space. In addition, a second virtual line called the bar marks the 2-gigabyte address. The bar separates storage below the 2-gigabyte address, called below the bar, from storage above the 2-gigabyte address, called above the bar. The area above the bar is intended for data; no programs can run above the bar. IBM has reserved an area of storage above the bar for special use to be developed in the future. For an application program to use virtual storage above the bar, a program must request storage above the bar, be in AMODE64, and use the new z/Architecture Assembler instructions. The 64-Bit Virtual Address Space is illustrated in Figure 1.

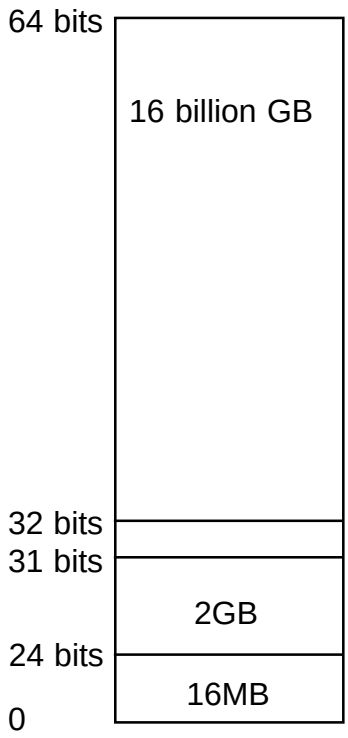

Figure 1: 64-Bit Virtual Address Space

Z/ARCHITECTURE PSW

The z/Architecture PSW is 128 bits in length (16 bytes) with bit 12 set to 0 (indicates a 128-bit PSW). The LPSW instruction loads an 8-byte PSW and extends it to the new 16-byte format. The LPSWE instruction can be used to set the full 128-bit PSW.

The z/OS supervisor remaps the 128-bit PSW to 64-bits in control blocks such as the TCB, RB, IHSA, and SDWA. Bits 33-

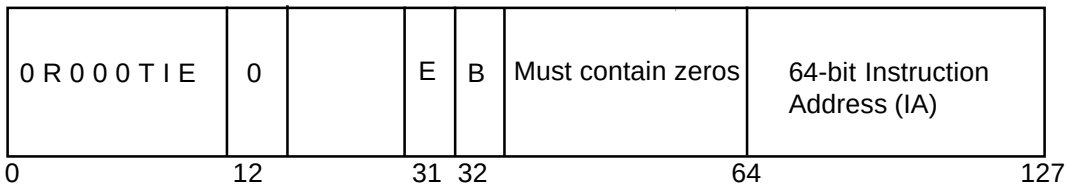

Figure 2: 128-bit PSW

96 are removed and bit-12 is set to 1. As one would expect, all PSWs that are stored in the PSA are 128-bit PSWs.

Figure 2 shows a 128-bit PSW.

Bit 12 – 0 indicates a 128-bit PSW

Bit 31 (Extended Addressing Mode)

Bit 32 (Basic addressing mode).

EB meaning:

00 - 24-bit Addressing Mode

01 - 31-bit Addressing Mode

11 - 64-bit Extended Addressing Mode

10 - Specification exception.

Bit 127 = Must be zero.

ADDRESS SPACE CONTROL ELEMENT

New with z/Architecture is the Address Space Control Element (ASCE), which describes an address space. There are two types of address space:

- A virtual address space, which is described by translation tables.
- The Real Space, which has no translation tables, because a virtual address is translated to the same identical real address.

The ASCE resides in the following control registers:

- Control Registser 1 (CR1) – Primary Address Space Control Element (PASCE).
- Control Registser 7 (CR7) – Secondary Address Space Control Element (SASCE).
- Control Registser 13 (CR13) – Home Address Space Control Element (HASCE).

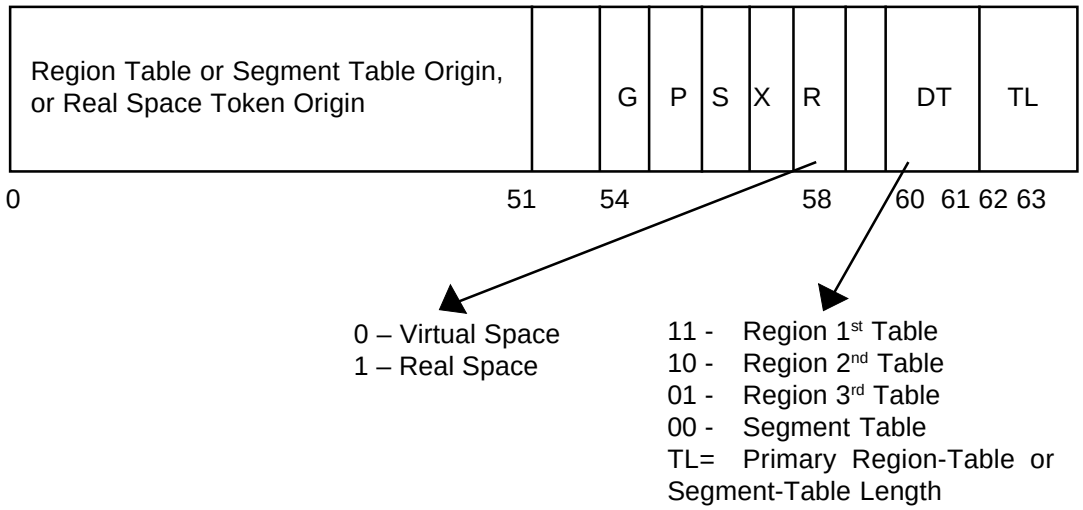

Figure 3: ASCE

This is illustrated in Figure 3.

When bit 58 of the ASCE is 1, the Primary Real Space is designated and the effective virtual address is translated to itself as a real address. Addresses in the Real Space are virtual addresses, but they just translate to the identical real address.

A new translation exception code, the ASCE Type Exception (X'38'), has been introduced, and will occur under the following conditions:

- Virtual address > 2**31-1 and ASCE DT bits (60-61) = 00
- Virtual address > 2**42-1 and ASCE DT bits (60-61) = 01
- Virtual address > 2**53-1 and ASCE DT bits (60-61) = 10.

64-BIT VIRTUAL ADDRESSING

In z/Architecture, there are up to five levels of translation tables:

- Region First (indexed by RFX)
- Region Second (indexed by RSX)

- Region Third (indexed by RTX)
- Segment (indexed by SX)
- Page (indexed by PX).

The number of translations depends on the size of the address space. The segment and page tables are always present and

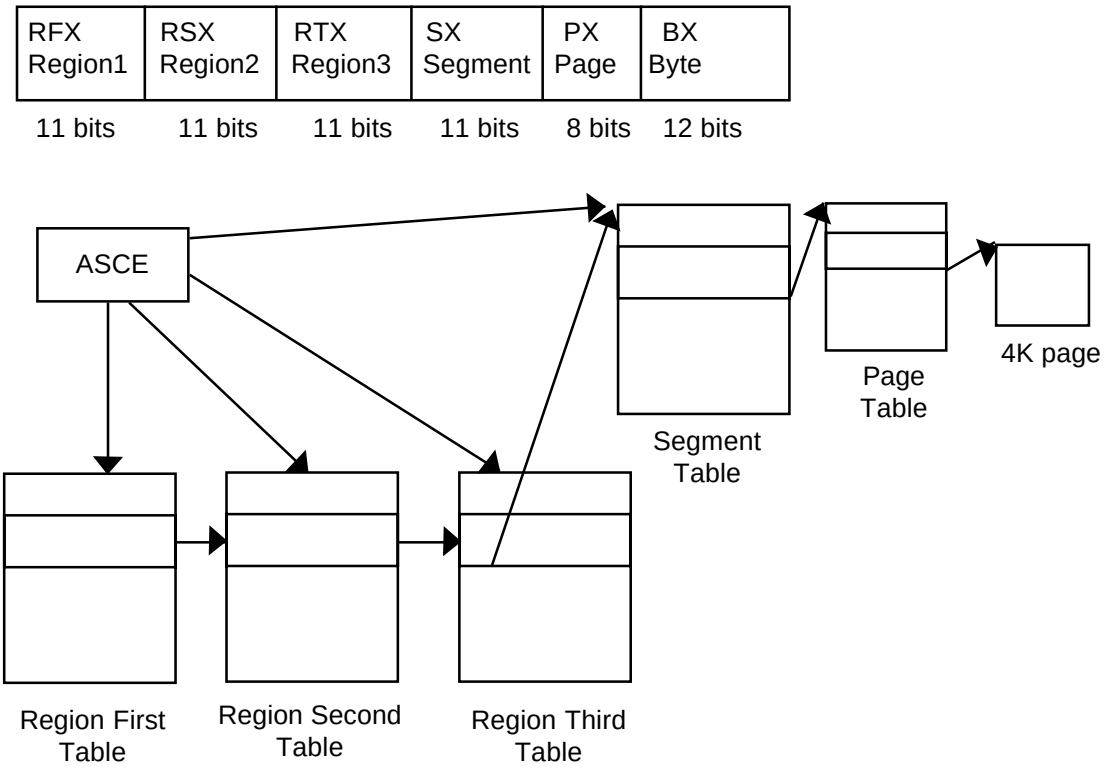

Figure 4: 64-bit effective Virtual Address translation overview

always used. The Region Tables are present and are used only if it is necessary. The starting point of the translation is designated in the Address Space Control Element (ASCE), bits 60 to 61(ASCE.DT). When the ASCE used in a translation is a Region-First-Table designation, the translation process consists of a five-level look-up using the five tables described above. All these tables can reside in real or absolute storage. When the ASCE is a Region-SecondTable designation, Region-Third-Table designation, or segment table designation, the look-ups in the

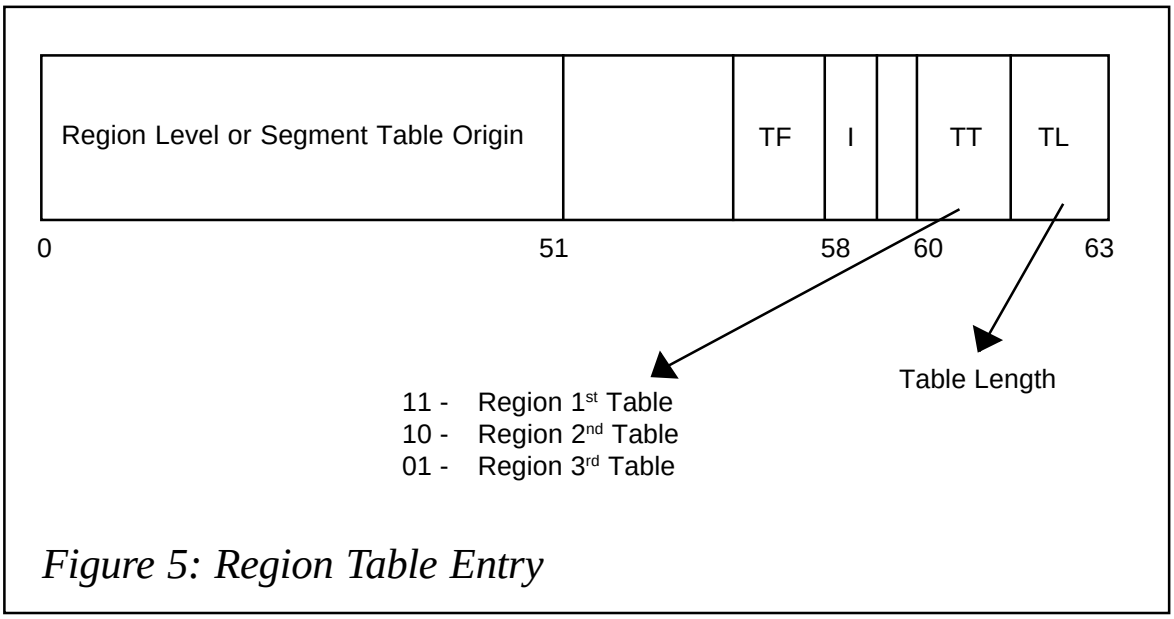

Figure 5: Region Table Entry

levels of tables above the designated level are omitted.

Note: Region Tables are not supported in OS/390 R10 and z/OS V1R1.

An overview of 64-bit effective Virtual Address translation is shown in Figure 4.

Region Table Entry

The Region First, Second, and Third Table Entries all have the same format – see Figure 5.

The following Region Translation Exceptions codes will occur

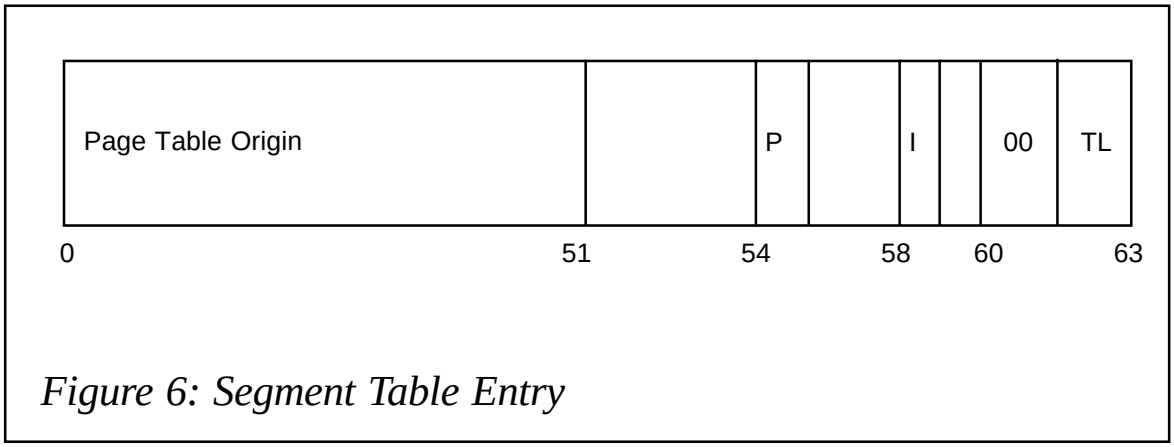

Figure 6: Segment Table Entry

under the following conditions:

- Region First Translation Exception – X'39'.

  36-53 – the I bit in the Region First Table Entry = 1 and the entry is indexed by RFX.

- Region Second Translation Exception – X'3A'.

  37-53 – the I bit in the Region Second Table Entry = 1 and the entry is indexed by RSX .

- Region Third Translation Exception – X'3B'.

  38-53 – the I bit in the Region Third Table Entry = 1 and the entry is indexed by RTX.

Segment Table Entry

A Segment Table Entry is illustrated in Figure 6.

Page Table Entry

The Page Table Entry in Z/Architecture is like the ESA/390 format but with the real frame address extended on the left to support 64-bit addresses. This is illustrated in Figure 7.

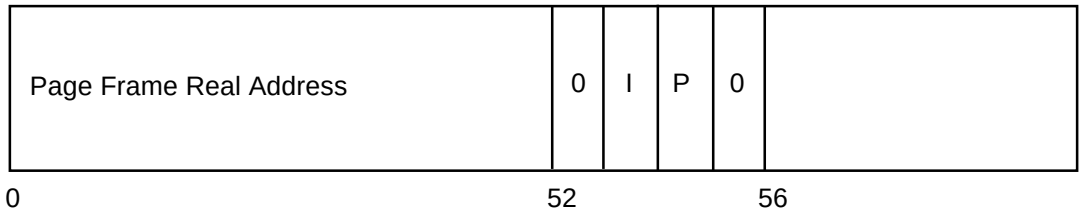

Figure 7: Page Table Entry

THE PREFIX STORAGE AREA

The Prefix Storage Area (PSA) resides in two 4KB frames on an 8KB boundary. Low address protection is applied to the first 512 bytes (0-511) and bytes 4096 through to 4607.The NEW and

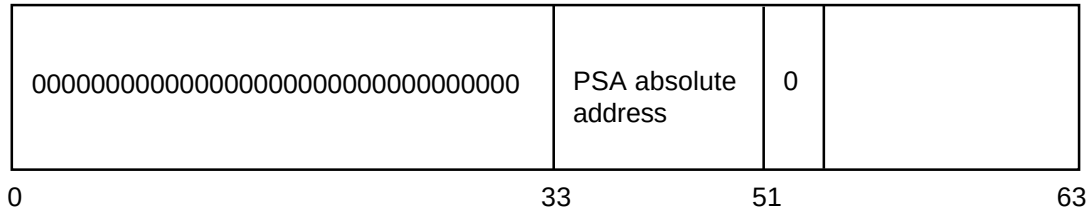

Figure 8: Prefix Register

OLD PSWs have been moved and are now 16 bytes in length. The ESA/390 OLD/NEW PSW area is reserved (24-127). The ESA/390 Machine Check and Store Status areas are remapped as the z/Architecture OLD/NEW PSW area (288-511).

Two new PSA macro definitions have been defined:

- IHAPSAE – PSA Extension (PSAE). The PSAE maps the ESAME format of the first page of the PSA.
- IHAPSAX – PSA Extension (PSAX). The PSAX maps the architected second page of the PSA.

PREFIX REGISTER

The Prefix Register is now 64 bits wide, but only bits 33-50 are used. Bits 0-32 must be zero. The SIGP Set-Architecture to z/Architecture order sets bit 51 to 0. It is illustrated in Figure 8.

GENERAL PURPOSE REGISTERS

The 16 General Purpose Registers (GPRs) in Z/Architecture mode are 64 bits long and are numbered 0-63. The right-half of the registers (the old 32-bit part) are now numbered 32-63. The ESA/390 instructions will use only the right halves of a 64-bit register. An important concept to understand is that you can use the 64-bit registers without being in 64-bit address mode.

A register is treated as 64-bits for:

- Address generation in 64-bit mode
- GPR operands of non-modal 64-bit instructions
- GPR operands of modal instructions in 64-bit mode.

A register is treated as 32 bits for:

- Address generation in 24/31-bit modes
- GPR operands of non-modal 32-bit instructions
- GPR operands of modal instructions in 24/31-bit mode.

Note: modal and non-modal instructions will be explained in *z/Architecture Instruction Set*.

CONTROL REGISTERS

All the 16 Control Registers are implemented as 64 bits. Control Registers 1, 7, 10, 11, 13, and 15 are treated as 64-bit addresses.

- CR1= Primary ASCE
- CR7= Secondary ASCE
- CR10= PER Range Start
- CR11= PER Range End
- CR12= Trace Entry Address
- CR13= Home ASCE
- CR15= Linkage Stack Designator.

Two new instructions, LCTLG (LOAD CONTROL) and STCTG (STORE CONTROL), have been introduced, which operate on the full 64-bit control registers.

ACCESS REGISTERS

All 16 Access Registers in z/Architecture are still 32 bits wide. The Access Register Translation (ART) is logically unchanged from ESA/390. There are minor adjustments, mainly due to the

use of the ASCE rather than the Segment-Table Designation (STD) and to the different mapping of the Address Space Second Table Entry (ASTE).

SET ARCHITECTURE SIGP ORDER

A new SIGP order, Set Architecture, has been defined that allows an operating system to switch into z/Architecture mode. This is normally performed at IPL time. For each CPU that changes from ESA/390 to z/Architecture, the 8-byte PSW for each CPU in the configuration is changed to a 16-byte PSW.

The PSW bits are set as follows:

- Bits 0-11 and bits 13-32 are set equal to the same bits of the 8-byte PSW.
- Bit 12 and bits 33-96 are set to zero.
- Bits 97-127 are set equal to bits 33-63 of the 8-byte PSW.

STORE FACILITY LIST

The Store Facility List (STFL) instruction is a privileged instruction which stores a list of bits describing installed facilities at location 200-203 (PSA). The STFL instruction has been added to the ESA/390 instruction set. It's illustrated in Figure 9.

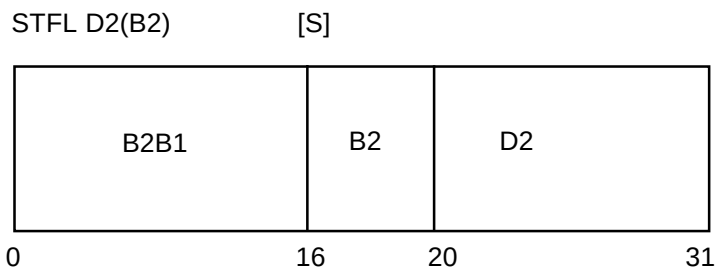

Figure 9: The Store Facility List instruction

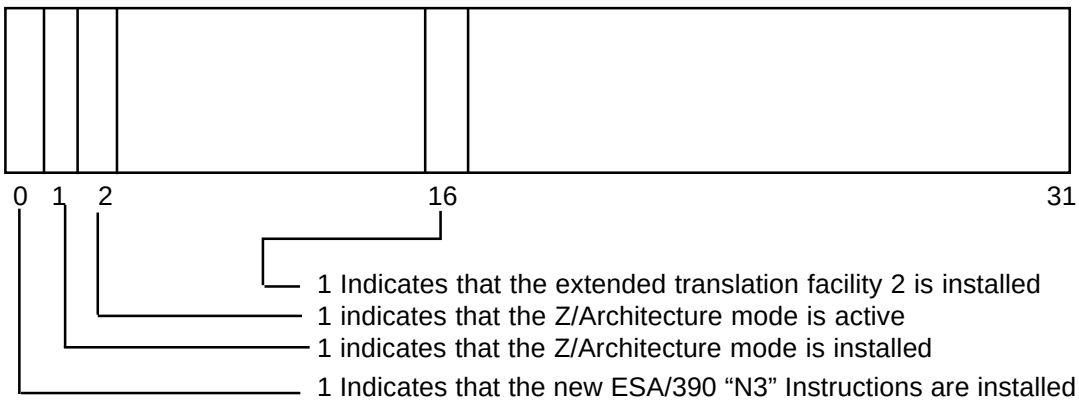

Figure 10: Definitions

The bits are defined as shown in Figure 10.

This article will be concluded next month.

R F Perretta
Systems Consultant
Millenium Computer Consultancy (UK)

Micro Focus has announced the latest version of EnterpriseLink, its legacy-to-Web transformation and integration solution, which provides Web services to legacy applications.

The updated product provides immediate Web services access for clients built with Microsoft .NET Studio and allows reuse of legacy assets in a low-risk high-ROI environment.

By providing a Web services interface, EnterpriseLink now enables companies to establish an XML-based, language-neutral connection to green screen mainframe applications. As a result, companies can get their legacy applications up and running on the Web or other client/server platforms quickly.

EnterpriseLink is deployable in Unix and Windows environments to move legacy applications to Web services more quickly than with manual recoding.

For further information contact:
Micro Focus, Old Bath Road, Newbury, Berks RG14 1QN, UK.
Tel: (01635) 32646.
URL: http://www.microfocus.com/products/enterpriselink/.

* * *

IBM has announced Enterprise Developer Server for z/OS Version 5.0, which provides the runtime libraries for programs that were developed with either WebSphere Studio Enterprise Developer V5.0 or VisualAge Generator Developer and execute on z/OS. These libraries provide common runtime subroutines that are shared by all Enterprise Generation Language (EGL) programs created with WebSphere Studio Enterprise Developer, such as data conversion and error management.

WebSphere Studio Enterprise Developer V5.0 brings J2EE, Rapid Application Development (RAD), and team support to diverse enterprise application development organizations; it embraces a component reuse model-based development paradigm.

For further information contact your local IBM representative.
URL: http://www.ibm.com.

* * *

IBM has released Tivoli System Automation for OS/390 (SA OS/390) under the Tivoli Environment-Managed Licensing Model, which means pricing and licensing are based on what is managed rather than how the software is implemented.

The software is designed to automate I/O, processor, and system operations and includes canned automation for IMS, CICS, IBM Tivoli Workload Scheduler, and DB2. Key functions include Parallel Sysplex application automation, policy-based self-healing, integration, processor operations (ProcOps) and I/O operations, and SAP R/3 high-availability automation.

Other features include cluster-wide policy to help reduce complexity, implementation time, coding, and support plus Parallel Sysplex management and automation functions, including single system image, single point of control, and Parallel Sysplex application automation.

For further information contact your local IBM representative.
URL: http://www.tivoli.com/products