145

MVS update

October 1998

In this issue

MVS Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 33598
From USA: 01144 1635 33598
E-mail: xephon@compuserve.com

North American office

Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75067
USA
Telephone: 940 455 7050

Contributions

If you have anything original to say about MVS, or any interesting experience to recount, why not spend an hour or two putting it on paper? The article need not be very long – two or three paragraphs could be sufficient. Not only will you be actively helping the free exchange of information, which benefits all MVS users, but you will also gain professional recognition for your expertise, and the expertise of your colleagues, as well as some material reward in the form of a publication fee – we pay at the rate of £170 ($250) per 1000 words for all original material published in *MVS Update*. If you would like to know a bit more before starting on an article, write to us at one of the above addresses, and we'll send you full details, without any obligation on your part.

Editor

Jaime Kaminski

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.

MVS Update on-line

Code from *MVS Update* can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label.

Subscriptions and back-issues

A year's subscription to *MVS Update*, comprising twelve monthly issues, costs £325.00 in the UK; $485.00 in the USA and Canada; £331.00 in Europe; £337.00 in Australasia and Japan; and £335.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1992 issue, are available separately to subscribers for £29.00 ($43.00) each including postage.

Printed in England.

OS/390 Version 2 Release 6

INTRODUCTION

25 September 1998 saw the general release of OS/390 Version 2 Release 6, six months after the release of Version 2 Release 5 (see *MVS Update* Issue 140 May 1998).

VERSION 2 RELEASE 6

The major additions and enhancements to Release 6 include:

- WebSphere Application Server for OS/390.
- Performance improvements to OS/390 Unix.
- eNetwork Communications Server performance and scalability enhancements.
- New Network File System function.
- Integration of Novell's NDS network directory services.
- Distributed File Service (DFS) enhancements.
- Support for IEEE floating point.
- Full support by the Tivoli framework.
- Further enhancements to RACF, LDAP, ICSF, and Firewall Technologies security.

ANALYSIS

The latest release of OS/390 continues the same logical progression found in previous releases. The operating system is being promoted as a major platform for server consolidation, electronic commerce, application enablement, and business intelligence.

Clustering and e-business

Parallel Sysplex clustering improvements are being promoted as a vital component for e-business. This is a function of the 99.999% availability and near limitless capacity, via Coupling Facility administration improvements. In addition, there is a Resource Measurement Facility for on-line monitoring of Coupling Facility activity, and an enhanced Workload Manager, which can now prioritize and manage all WebSphere Application Server Web requests, both SSL and non-SSL, within a Parallel Sysplex cluster.

TCP/IP enhancements

New Web-serving capabilities include the integrated WebSphere Application Server for OS/390, which is actually an uprated and renamed Domino Go Webserver. This includes automated support of digital certificates and better Java servlet support via a new servlet engine. Digital certificate support comes via OS/390, now acting as the local certificate authority to issue and manage certificates locally.

The eNetwork Communications Server has new TCP/IP services to exploit the enhancements in Version 2 Release 5, providing a single IP stack for all customer and TCP/IP communications. Figures provided by IBM suggest that the enhanced TCP/IP services have shown a performance improvement of up to 15 times for a full range of applications including file transfer, Telnet, Unix, or any ISV application using the TCP/IP sockets interface.

Server consolidation

Server consolidation is a crucial element of OS/390 strategy, as seen with Release 1 of Novell Network Services for OS/390. This will facilitate the consolidation of the numerous Novell servers into a single machine.

Support for NDS means the mainframe can become a full-blown enterprise directory server. In December a beta version of Component Broker for OS/390, for Releases 5 and 6, will be released.

Better LDAP support allows users running programs on OS/390 Unix to enter and extract data from any LDAP directory service that accepts

LDAP Version 2 or 3 protocols. Multiple LDAP servers can be operated independently on a single OS/390 image.

Application support

Version 2 Release 6 sees Component Broker for OS/390 made available as part of a beta program for Version 2 Release 5 and later users. On a more concrete basis, there are performance improvements in OS/390 Unix for porting Unix software to System/390s. Also, the NFS has been rewritten, improving performance and reducing CPU time for an NFS operation by up to 50%. Performance throughput is claimed to have been improved by up to 75% for an individual application read-and-write operation to the NFS. There is now support for the latest Sun NFS Version 3. Support for IEEE floating point is implemented in the C/C++ compiler, for porting applications that use IEEE floating point from other servers to the System/390. The function has already been incorporated into the G5 Server. Also new on the comms front is support for Unix Sendmail, for consolidating e-mail management and administration on the System/390.

Open Systems Adapter 2

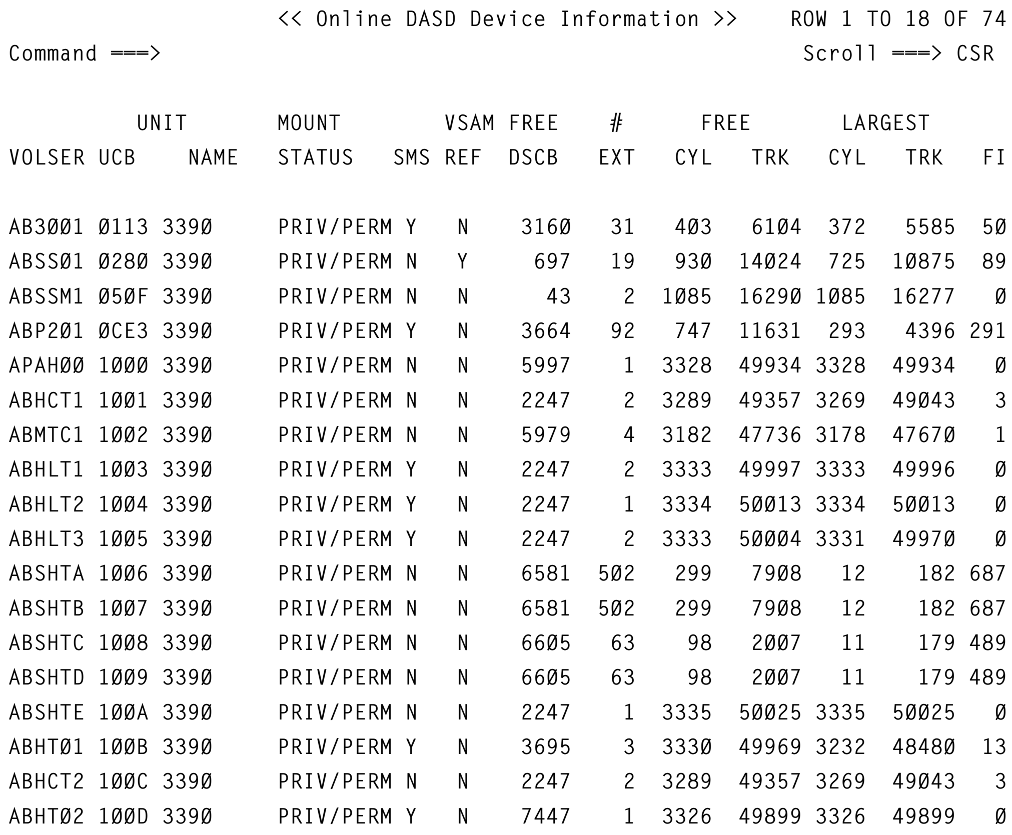
A considerably improved System/390 Open Systems Adapter 2 (OSA-2) is provided for LAN connections, providing IP multicast support for TCP/IP. All OSA-2 features can now accept LAN traffic supporting new router protocols that use multicast addressing such as Open Shortest Path First (OSPF). Also, Novell Network Services for OS/390 is supported exclusively by the Fast Ethernet OSA-2 feature for IPX users. This will provide directory support for new and existing Novell users, enable server consolidation, and allow for centralized administration.

Also announced were OSA/SF enhancements for TCP/IP applications, for improving reliability with multiple IP addresses for each unit address pair and providing a redundant path with designation of a primary or a secondary default OSA Address Table (OAT) entry.

ISPF display of UCB information

INTRODUCTION

Displaying device information is a basic task that most systems programmers and storage administrators perform on a daily basis. With the changes that have occurred within MVS and SMS over the years, new services have been provided to make it easier to obtain the desired information about devices. A routine that we developed and put in our toolkit is an ISPF dialog called $DSPACE. $DSPACE as it is designed will display information about on-line DASD devices. $DSPACE utilizes a routine that we have included called $UCBINFO. $UCBINFO as called by $DSPACE returns only DASD information. It returns information about dynamic devices as well as UCBs that are above the 16M line.



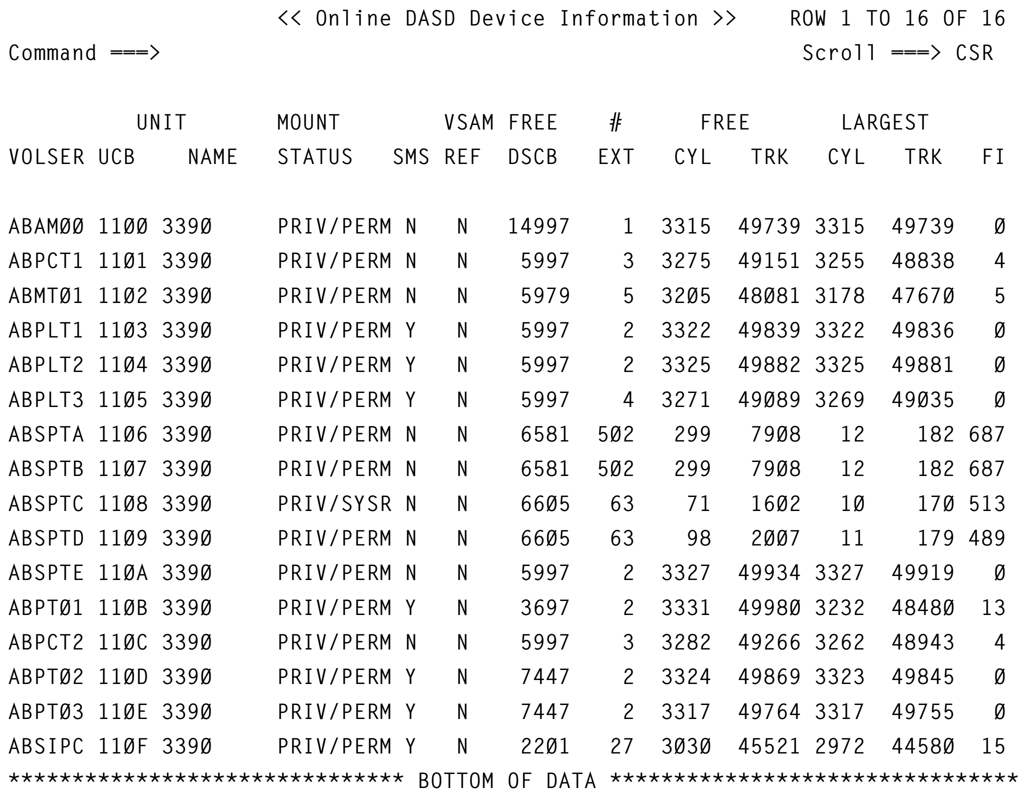
```
                      << Online DASD Device Information >>    ROW 1 TO 18 OF 74
Command ===>                                                  Scroll ===> CSR

         UNIT        MOUNT        VSAM FREE    #      FREE       LARGEST
VOLSER UCB    NAME   STATUS    SMS REF  DSCB   EXT   CYL    TRK   CYL    TRK   FI


AB3ØØ1 Ø113 339Ø     PRIV/PERM Y    N    316Ø   31   4Ø3   61Ø4   372   5585  5Ø
ABSSØ1 Ø28Ø 339Ø     PRIV/PERM N    Y     697   19   93Ø  14Ø24   725  1Ø875  89
ABSSM1 Ø5ØF 339Ø     PRIV/PERM N    N      43    2  1Ø85  1629Ø 1Ø85  16277   Ø
ABP2Ø1 ØCE3 339Ø     PRIV/PERM Y    N    3664   92   747  11631   293   4396 291
APAHØØ 1ØØØ 339Ø     PRIV/PERM N    N    5997    1  3328  49934 3328  49934   Ø
ABHCT1 1ØØ1 339Ø     PRIV/PERM N    N    2247    2  3289  49357 3269  49Ø43   3
ABMTC1 1ØØ2 339Ø     PRIV/PERM N    N    5979    4  3182  47736 3178  4767Ø   1
ABHLT1 1ØØ3 339Ø     PRIV/PERM Y    N    2247    2  3333  49997 3333  49996   Ø
ABHLT2 1ØØ4 339Ø     PRIV/PERM Y    N    2247    1  3334  5ØØ13 3334  5ØØ13   Ø
ABHLT3 1ØØ5 339Ø     PRIV/PERM Y    N    2247    2  3333  5ØØØ4 3331  4997Ø   Ø
ABSHTA 1ØØ6 339Ø     PRIV/PERM N    N    6581  5Ø2   299   79Ø8    12    182 687
ABSHTB 1ØØ7 339Ø     PRIV/PERM N    N    6581  5Ø2   299   79Ø8    12    182 687
ABSHTC 1ØØ8 339Ø     PRIV/PERM N    N    66Ø5   63    98   2ØØ7    11    179 489
ABSHTD 1ØØ9 339Ø     PRIV/PERM N    N    66Ø5   63    98   2ØØ7    11    179 489
ABSHTE 1ØØA 339Ø     PRIV/PERM N    N    2247    1  3335  5ØØ25 3335  5ØØ25   Ø
ABHTØ1 1ØØB 339Ø     PRIV/PERM Y    N    3695    3  333Ø  49969 3232  4848Ø  13
ABHCT2 1ØØC 339Ø     PRIV/PERM N    N    2247    2  3289  49357 3269  49Ø43   3
ABHTØ2 1ØØD 339Ø     PRIV/PERM Y    N    7447    1  3326  49899 3326  49899   Ø
```

Figure 1: Sample display from TSO $DSPACE

We structured $DSPACE so that we could either obtain information about all on-line DASD devices, or screen the information we want to see by either a volser pattern or a UCB address pattern, ie:

- TSO $DSPACE – return all DASD devices (see Figure 1)
- TSO $DSPACE SPE – return all volsers starting with SPE
- TSO $DSPACE U=11– return all volsers with a UCB starting with 11 (see Figure 2).

The device-type information that is displayed is not the actual device type, but is the device type for the UCB that is obtained from the eligible device table. You may wish to change that to display the actual device. Several macros that we use to ease program development have been included for your reference.



```
                     << Online DASD Device Information >>    ROW 1 TO 16 OF 16
Command ===>                                                  Scroll ===> CSR

          UNIT       MOUNT        VSAM FREE     #      FREE       LARGEST
VOLSER UCB    NAME   STATUS   SMS REF  DSCB   EXT   CYL   TRK   CYL   TRK   FI

ABAM00 1100 3390     PRIV/PERM N   N   14997    1  3315  49739 3315  49739   0
ABPCT1 1101 3390     PRIV/PERM N   N    5997    3  3275  49151 3255  48838   4
ABMT01 1102 3390     PRIV/PERM N   N    5979    5  3205  48081 3178  47670   5
ABPLT1 1103 3390     PRIV/PERM Y   N    5997    2  3322  49839 3322  49836   0
ABPLT2 1104 3390     PRIV/PERM Y   N    5997    2  3325  49882 3325  49881   0
ABPLT3 1105 3390     PRIV/PERM Y   N    5997    4  3271  49089 3269  49035   0
ABSPTA 1106 3390     PRIV/PERM N   N    6581  502   299   7908   12    182 687
ABSPTB 1107 3390     PRIV/PERM N   N    6581  502   299   7908   12    182 687
ABSPTC 1108 3390     PRIV/SYSR N   N    6605   63    71   1602   10    170 513
ABSPTD 1109 3390     PRIV/PERM N   N    6605   63    98   2007   11    179 489
ABSPTE 110A 3390     PRIV/PERM N   N    5997    2  3327  49934 3327  49919   0
ABPT01 110B 3390     PRIV/PERM Y   N    3697    2  3331  49980 3232  48480  13
ABPCT2 110C 3390     PRIV/PERM N   N    5997    3  3282  49266 3262  48943   4
ABPT02 110D 3390     PRIV/PERM Y   N    7447    2  3324  49869 3323  49845   0
ABPT03 110E 3390     PRIV/PERM Y   N    7447    2  3317  49764 3317  49755   0
ABSIPC 110F 3390     PRIV/PERM Y   N    2201   27  3030  45521 2972  44580  15
****************************** BOTTOM OF DATA ********************************
```

Figure 2: Sample display TSO $DSPACE U=110

$DSPACE

```
         TITLE '$DSPACE - DISPLAY ON-LINE DASD INFORMATION'
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
* CSECT   : $DSPACE                                                   *
* MODULE  : $DSPACE                                                   *
* DESC    : $DSPACE IS A PROGRAM THAT IS DESIGNED TO RUN UNDER ISPF.  *
*           IT IS USED TO DISPLAY INFORMATION ABOUT ON-LINE DASD. AN  *
*           ISPF TABLE IS USED TO STORE AND DISPLAY THE INFORMATION.  *
*           IT WILL DISPLAY ALL ON-LINE DEVICES, OR YOU CAN SCREEN     *
*           WHICH DEVICES YOU ARE INTERESTED IN BY VOLUME SERIAL, OR  *
*           BY THE UCB ADDRESS.                                       *
* MACROS  : $PFPRO $PFEPI $PFSTG LSPACE LINK DEVTYPE EDTINFO          *
*           IEFUCBOB IECSDSL1 $CALL                                   *
* DSECTS  : UCB_STR1 UCB_STR2 UCBDSECT MYF4                           *
* INPUT   : NONE                                                      *
* OUTPUT  : NONE                                                      *
* PLIST   : PARAMETERS ARE PASSED IN THE TSO CPPL STRUCTURE           *
*           POSSIBLE SETTINGS ARE : NO PARMS, DISPLAY ALL DASD        *
*           U=**** WHERE **** IS ONE TO 4 DIGITS OF THE UCB           *
*           ****** WHERE ****** IS 1 TO 6 CHARACTERS OF THE VOLUME    *
*                        SERIAL OF THE DESIRED UNITS                  *
* CALLS   : ISPLINK $UCBINFO                                          *
* NOTES   : VALIDATED UNDER MVS 5.2.2 WITH DFSMS/MVS 1.3              *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         EJECT
$DSPACE  $PFPRO R12,AM=31,RM=ANY
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  PICK UP THE CPPL AND SEE WHAT WE HAVE BEEN PASSED                  *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         MVI   SCREEN,C'N'             SET THE INITIAL SCREEN CONTROL
         L     R2,Ø(R1)                POINT AT THE CPPL
         LH    R3,2(R2)                GET LENGTH OF ACTUAL DATA
         LA    R3,4(R3,R2)             POINT TO BEGINNING OF PARMS
         LH    R4,Ø(R2)                GET LENGTH OF CPPL
         LA    R4,Ø(R4,R2)             CALCULATE END POINT
         SR    R4,R3                   CALCULATE LENGTH OF PARMS
         BZ    NO_PARMS                BRANCH IF NO PARMS PRESENT
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  USER IS ASKING FOR SOMETHING.  WE NEED TO DETERMINE IF THEY WANT   *
*  TO SCREEN OFF BY UCB DESIGNATION, OR DO THEY WANT TO SCREEN OFF    *
*  BY THE VOLUME SERIAL.                                              *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         CH    R4,HALF2                Q. PARMS 2 BYTES OR LESS
         BNH   ISO_VOL                 A. NO, BYPASS UCB CHECK
CHK_MORE DS    ØH
         CLC   UCB_SC,Ø(R3)            Q. FIRST 2 BYTES  U=
         BNE   ISO_VOL                 A. NO, VOLUME SCREEN ASSUMED
         SH    R4,HALF3                ADJUST THE LENGTH
         STH   R4,SC_SIZE              SAVE IT
         LA    R3,2(,R3)               ADJUST POINTER
         EX    R4,MVE_PARM             GO MOVE THE DATA
         MVI   SCREEN,C'U'             INDICATE UCB SCREENING
         B     NO_PARMS                BRANCH ON DOWN
ISO_VOL  DS    ØH
         BCTR  R4,Ø                    DECREMENT LENGTH
         STH   R4,SC_SIZE              SAVE THE LENGTH
         EX    R4,MVE_PARM             GO MOVE THE DATA
         MVI   SCREEN,C'V'             INDICATE VOLUME SCREENING
NO_PARMS DS    ØH
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  SET UP WITH ISPF, DEFINE THE VARIABLES AND OUR TABLE              *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         $CALL @ISPLINK,                                              +
               (VDEFINE,VAR_SPF,VOL,TYP_SPF,VARL_SPF,LIST),            +
               VL,MF=(E,@CALL)
         $CALL @ISPLINK,                                              +
               (TBCREATE,PAN_NAME,,NAM_SPF,NOWRITE,REPLACE),           +
               VL,MF=(E,@CALL)
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  LINK TO $UCBINFO TO GET THE INFO WE WILL PROCESS                   *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         LINK EP=$UCBINFO,                                             +
               PARAM=(UCB_TYPE,@FCHUNK),                               +
               VL,MF=(E,@CALL)
         LTR   R15,R15
         BNZ   TB_END
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  ESTABLISH SOME BASE REGISTERS FOR DATA ACCESS                      *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         USING UCB_STR1,R2             TELL ASSEMBLER
         USING UCB_STR2,R4             TELL ASSEMBLER
         USING UCBDSECT,R5             TELL ASSEMBLER
         L     R2,@FCHUNK              GET @(FIRST CHUNK)
CHUNK_O  DS    ØH
         LR    R4,R2                   GET @(CURRENT CHUNK)
         L     R3,UCB_##               NUMBER OF ENTRIES IN THE CHUNK
         BCTR  R3,Ø                    DECREMENT THE COUNT
         LA    R4,D_SIZE(,R4)          BUMP THE POINTER
         LA    R5,4(,R4)               POINT TO THE UCB COPY
CHUNK_I  DS    ØH
         TM    UCBSTAT,UCBONLI         Q. IS THIS DEVICE ON-LINE
         BNO   BUMP_I                  MOVE TO NEXT ENTRY
         CLI   SCREEN,C'N'             Q. NEED TO SCREEN
         BE    NO_SCRE                 A. NO, BRANCH
         CLI   SCREEN,C'V'             Q. VOLSER SCREEN
         BNE   UCB_SCRE                A. NO
         LA    R8,UCBVOLI              GET @(VOLSER)
         B     COM_SCRE                BRANCH TO COMMON CODE
UCB_SCRE DS    ØH
         LA    R8,UCB_UNIT             GET @(CHARACTER UCB)
COM_SCRE DS    ØH
         LH    R9,SC_SIZE              GET THE SIZE OF THE SCREEN ARG
         EX    R9,CMP_PARM             GO DO THE COMAPRE
         BNE   BUMP_I                  NO MATCH
NO_SCRE  DS    ØH
         MVC   VOL,UCBVOLI             GET THE VOLID
         MVC   UCB,UCB_UNIT            GET THE ADDRESS
         MVC   W_UCB,UCB_UNIT          GET THE ADDRESS
         MVC   ST(4),UNKNOWN           SET INITIAL STATUS
         TM    UCBSTAB,UCBBPRV         Q. PRIVATE
         BNO   CHK_PUB                 A. NO, SEE IF PUBLIC
         MVC   ST(4),PRIVATE           SET TO PRIVATE
         B     ST_SET                  BRANCH DOWN
CHK_PUB  DS    ØH
         TM    UCBSTAB,UCBBPUB         Q. PUBLIC
         BNO   CHK_STG                 A. NO, SEE IF STORAGE
         MVC   ST(4),PUBLIC            SET TO PUBLIC
         B     ST_SET                  BRANCH DOWN
CHK_STG  DS    ØH
         TM    UCBSTAB,UCBBSTR         Q. STORAGE
         BNO   ST_SET                  A. NO, BRANCH DOWN
         MVC   ST(4),STORAGE           SET TO STORAGE
ST_SET   DS    ØH
         MVI   ST+4,C'/'               MOVE IN THE DELIMETER
         TM    UCBSTAT,UCBPRES         Q. VOL PERMANENTLY RESIDENT
         BNO   CK_SYSR                 A. NO, NEXT CHECK
         MVC   ST+5(4),PERM            MOVE IN VALUE
CK_SYSR  DS    ØH
         TM    UCBSTAT,UCBSYSR         Q. RES VOL
         BNO   NOT_SYSR                A. NO, BRANCH
         MVC   ST+5(4),SYSR            MOVE IN VALUE
NOT_SYSR DS    ØH
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  USE LSPACE TO OBTAIN VOLUME INFORMATION                            *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         MVC   @LSPACE(L_LSPACE),M_LSPACE PRIME THE AREA
*
         LSPACE UCB=(R5),                                              +
               DATA=@LSPACEW,                                          +
               F4DSCB=@F4DSCB,                                         +
               MF=(E,@LSPACE)
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  PROCESS THE DATA RETURNED BY LSPACE                                *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         LA    R15,@LSPACEW            GET @(RETURNED AREA)
         USING @LSPACED,R15            ESTABLSIH THE BASE
         L     R7,LSPDNEXT             GET NUMBER OF EXTENTS
         CVD   R7,D_WORK               CONVERT IT TO DECIMAL
         MVC   C_WORK,E_PL8            MOVE IN THE EDIT PATTERN
         ED    C_WORK,D_WORK+4         EDIT THE DATA
         MVC   EXT#,C_WORK+L'C_WORK-L'EXT# MOVE IN CHARACTERS
         L     R7,LSPDTCYL             GET NUMBER OF FREE CYLINDERS
         CVD   R7,D_WORK               CONVERT IT TO DECIMAL
         MVC   C_WORK,E_PL8            MOVE IN THE EDIT PATTERN
         ED    C_WORK,D_WORK+4         EDIT THE DATA
         MVC   FCYL,C_WORK+L'C_WORK-L'FCYL MOVE IN CHARACTERS
         L     R7,LSPDTCYL             GET NUMBER OF FREE CYLINDERS
         MH    R7,TRKPERCY             CALCULATE NUMBER OF TRACKS
         A     R7,LSPDTTRK             GET NUMBER OF FREE CYLINDERS
         CVD   R7,D_WORK               CONVERT IT TO DECIMAL
         MVC   C_WORK,E_PL8            MOVE IN THE EDIT PATTERN
         ED    C_WORK,D_WORK+4         EDIT THE DATA
         MVC   FTRK,C_WORK+L'C_WORK-L'FTRK MOVE IN CHARACTERS
         L     R7,LSPDLCYL             GET NUMBER OF FREE CYLINDERS
         CVD   R7,D_WORK               CONVERT IT TO DECIMAL
         MVC   C_WORK,E_PL8            MOVE IN THE EDIT PATTERN
         ED    C_WORK,D_WORK+4         EDIT THE DATA
         MVC   LCYL,C_WORK+L'C_WORK-L'LCYL MOVE IN CHARACTERS
         L     R7,LSPDLCYL             GET NUMBER OF FREE CYLINDERS
         MH    R7,TRKPERCY             CALCULATE NUMBER OF TRACKS
         A     R7,LSPDLTRK             GET NUMBER OF FREE CYLINDERS
         CVD   R7,D_WORK               CONVERT IT TO DECIMAL
         MVC   C_WORK,E_PL8            MOVE IN THE EDIT PATTERN
         ED    C_WORK,D_WORK+4         EDIT THE DATA
         MVC   LTRK,C_WORK+L'C_WORK-L'LTRK MOVE IN CHARACTERS
         L     R7,LSPDFRAG             GET NUMBER OF FREE CYLINDERS
         CVD   R7,D_WORK               CONVERT IT TO DECIMAL
         MVC   C_WORK,E_PL8            MOVE IN THE EDIT PATTERN
         ED    C_WORK,D_WORK+4         EDIT THE DATA
         MVC   FIDX,C_WORK+L'C_WORK-L'FIDX MOVE IN CHARACTERS
         MVC   @DEVTYPE,ILIST_M        PRIME THE PARAMETER LIST
         DROP  R15
         LA    R15,@F4DSCB             GET @(FORMAT 4 DSCB)
         USING MYF4,R15                TELL THE ASSEMBLER
         LH    R7,DS4DSREC             GET NUMBER OF AVAILABLE DSCBS
         CVD   R7,D_WORK               CONVERT IT TO DECIMAL
         MVC   C_WORK,E_PL8            MOVE IN THE EDIT PATTERN
         ED    C_WORK,D_WORK+4         EDIT THE DATA
         MVC   DS#,C_WORK+L’C_WORK-L’DS# MOVE IN THE CHARACTERS
         ST    R5,@UCBADDR             STORE @(UCB) FOR DEVTYPE
         MVI   SI,C'N'                 SET INITIAL VALUE
         MVI   VS,C'N'                 SET INITIAL VALUE
         TM    DS4SMSFG,DS4SMS         Q. SMS MANAGED
         BNO   NOT_SMS                 A. BRANCH
         MVI   SI,C'Y'                 SET SMS ON
NOT_SMS  DS    ØH
         TM    DS4VSIND,DS4VSREF       Q. VSAM CAT. REF. THIS VOLUME
         BNO   NOT_VSAM                A. BRANCH
         MVI   VS,C'Y'                 SET VSAM ON
         DROP  R15
NOT_VSAM DS    ØH
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  USE THE DEVTYPE AND EDTINFO SERVICES TO DETERMINE WHAT THE DEVICE  *
*  TYPE IS.  KEEP IN MIND THAT THIS IS THE DEVICE TYPE AS KNOWN TO    *
*  ELIGIBLE DEVICE TABLE.                                             *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
```

```
         DEVTYPE ,(@DTYPE,L'@DTYPE),,,                                 +
               UCBLIST=(@UCBADDR,1,BELOW),                             +
               MF=(E,@DEVTYPE)
*
         EDTINFO RTNUNIT,                                              +
               DEVTYPE=@DTYPE,                                         +
               OUTUNIT=@EDTDATA,                                       +
               MF=(E,@EDTI)
*
         MVC   UN,@EDTDATA             MOVE IN THE UNIT NAME
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  NOW WE WANT TO TAKE THE CHARACTER REPRESENTATION OF THE UCB, AND   *
*  CREATE A WORKING VARIABLE WHICH WILL CONTAIN THE HEX VALUE OF THE *
*  UCB ADDRESS.  WE ARE DOING THIS SO THAT WE CAN THEN ASK ISPF TABLE *
*  SERVICES TO SORT THE UCBS IN ASCENDING ORDER.                    *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         TR    W_UCB,TR_TABLE          TRANSLATE THE CHARACTER DATA
         SR    R14,R14                 CLEAR REGISTER 14
         SR    R15,R15                 CLEAR REGISTER 15
         ICM   R14,B'1ØØØ',W_UCB+2     GET DIGIT
         ICM   R15,B'1ØØØ',W_UCB+3     GET DIGIT
         SRL   R14,2Ø                  SHIFT BY 4 BITS
         SRL   R15,24                  SHIFT BY 4 BITS
         OR    R14,R15                 LET'S PUT THEM TOGETHER
         STC   R14,WUCB+3              SAVE IT
*
         SR    R14,R14                 CLEAR REGISTER 14
         SR    R15,R15                 CLEAR REGISTER 15
         ICM   R14,B'1ØØØ',W_UCB       GET DIGIT
         ICM   R15,B'1ØØØ',W_UCB+1     GET DIGIT
         SRL   R14,2Ø                  SHIFT BY 4 BITS
         SRL   R15,24                  SHIFT BY 4 BITS
         OR    R14,R15                 LET'S PUT THEM TOGETHER
         STC   R14,WUCB+2              WUCB
*
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  ADD THE CURRENT INFO TO THE TABLE                                  *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*
         $CALL @ISPLINK,                                               +
               (TBADD,PAN_NAME),                                       +
               VL,MF=(E,@CALL)
*
BUMP_I   DS    ØH
*
         LA    R4,D_SIZE(,R4)          BUMP THE POINTER
         LA    R5,4(,R4)               POINT TO THE UCB COPY
         BCT   R3,CHUNK_I              PROCESS NEXT ENTRY ON CHUNK
         L     R2,UCB_FP               GET POINTER TO NEXT CHUNK
         LTR   R2,R2                   Q. IS THERE ANOTHER CHUNK
         BNZ   CHUNK_O                 A. YES, GO PROCESS IT
```

```
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  MOVE TO THE TOP OF THE TABLE                                       *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         $CALL @ISPLINK,                                               +
               (TBTOP,PAN_NAME),                                       +
               VL,MF=(E,@CALL)
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  SORT THE TABLE                                                     *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         $CALL @ISPLINK,                                               +
               (TBSORT,PAN_NAME,SORTLIST),                             +
               VL,MF=(E,@CALL)
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  NOW DISPLAY THE TABLE                                              *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         $CALL @ISPLINK,                                               +
               (TBDISPL,PAN_NAME,PAN_NAME),                            +
               VL,MF=(E,@CALL)
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  LET ISPF KNOW WE ARE DONE WITH THE TABLE                           *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
TB_END   DS    ØH
*
         $CALL @ISPLINK,                                               +
               (TBEND,PAN_NAME),                                       +
               VL,MF=(E,@CALL)
*
         $PFEPI
         TITLE '$DSPACE - LITERAL POOL, EXECUTABLE INSTRUCTIONS'
BIT_MASK EQU   B'11ØØØØØØ'             USED TO TEST PARMS IN CPPL
DS4VSREF EQU   X'8Ø'                   VALUE PER DFSMS 1.3
TRKPERCY DC    H'15'                   USED FOR TRACK CALCULATIONS
HALF2    DC    H'2'
HALF3    DC    H'3'
MVE_PARM MVC   SC_PAT(*-*),Ø(R3)       TARGET MOVE INSTRUCTION
CMP_PARM CLC   SC_PAT(*-*),Ø(R8)       TARGET COMPARE INSTRUCTION
UCB_SC   DC    CL2'U='                 USED TO CHECK FOR UCB SCREEN
UNKNOWN  DC    CL4'????’
PRIVATE  DC    CL4'PRIV’               PRIVATE USAGE
PUBLIC   DC    CL4'PUB '
STORAGE  DC    CL4'STOR'
UCB_TYPE DC    CL4'DASD'
PERM     DC    CL4'PERM'
SYSR     DC    CL4'SYSR'
NONO     DC    CL2'NN'
TR_TABLE DC    256XL1'ØØ'              TRANSLATION TABLE
         ORG   TR_TABLE+C'A'
         DC    XL6'ØAØBØCØDØEØF'
         ORG   TR_TABLE+C'Ø'
         DC    XL1Ø'ØØØ1Ø2Ø3Ø4Ø5Ø6Ø7Ø8Ø9'
         ORG
* THINGS USED/NEEDED FOR ISPF
```

```
PAN_NAME DC    CL8'$DSPACE'            NAME OF THE PANEL
TYP_SPF  DC    13CL8'CHAR',1CL8’FIXED' TYPE OF VARIABLES
NAM_SPF  DS    ØXL1
VAR_SPF  DC    C'(VOL,UCB,EXT#,FCYL,FTRK,LCYL,LTRK,FIDX,ST,UN,SI,VS,DS#+
               ,WUCB)'
VARL_SPF DS    ØF
         DC    A(L'VOL)
         DC    A(L'UCB)
         DC    A(L'EXT#)
         DC    A(L'FCYL)
         DC    A(L'FTRK)
         DC    A(L'LCYL)
         DC    A(L'LTRK)
         DC    A(L'FIDX)
         DC    A(L'ST)
         DC    A(L'UN)
         DC    A(L'SI)
         DC    A(L'VS)
         DC    A(L'DS#)
         DC    A(L'WUCB)
SORTLIST DC    C'(WUCB,N,A)'
E_PL8    DC    XL8'4Ø2Ø2Ø2Ø2Ø2Ø212Ø'   EDIT PATTERN
         TITLE '$DSPACE - MODEL PLIST FOR VARIOUS MACROS'
ILIST_M  DEVTYPE UCBLIST=(VARL_SPF,1,BELOW),INFOLIST=ILIST_D,MF=L
DT_RET_L EQU   *-ILIST_M               LET ASM CALC THE LENGTH
ILIST_D  DEVTYPE INFO=(DASD,DEVTYPE)
M_LSPACE LSPACE MF=L
L_LSPACE EQU   *-M_LSPACE
         TITLE '$DSPACE - DYNAMIC AREA'
         $PFSTG
@UCBADDR DS    F                       AREA FOR THE UCB FOR DEVTYPE
D_WORK   DS    D                       DOUBLEWORD WORK AREA
C_WORK   DS    CL8                     CHARACTER WORK AREA
@FCHUNK  DS    A                       @(FIRST CHUNK)
*
SC_SIZE  DS    H                       SIZE OF THE SCREEN PATTERN
SC_PAT   DS    CL6                     ACTUAL SCREEN PATTERN
SCREEN   DS    XL1                     FLAG TO INDICATE SCREENING
*
VOL      DS    CL6                     VOLUME SERIAL
UCB      DS    CL4                     4 CHARACTER UCB
EXT#     DS    CL4                     NUMBER OF FREE EXTENTS
FCYL     DS    CL4                     NUMBER OF FREE CYLINDERS
FTRK     DS    CL5                     NUMBER OF FREE TRACKS
LCYL     DS    CL4                     NUMBER OF FREE CYLINDERS
LTRK     DS    CL5                     NUMBER OF FREE TRACKS
FIDX     DS    CL3                     FRAGMENTATION INDEX
ST       DS    CL9                     VOLUME USE STATUS
UN       DS    CL8                     UNIT NAME FIELD
SI       DS    CL1                     SMS INDICATOR
VS       DS    CL1                     VSAM CATALOG REFERENCE
DS#      DS    CL5                     NUMBER OF DSCBS
```

```
WUCB     DS    XL4                      USED TO SORT UCBS
W_UCB    DS    XL4                      USED FOR UCB MANIPULATION
*
         DS    ØF                       FORCE ALIGNMENT
@DEVTYPE DEVTYPE MF=L                   EDT PARAMETER LIST
         DS    ØF                       FORCE ALIGNMENT
@DTYPE   DS    XL4Ø                     INFORMATION FROM DEVTYPE
*
@F4DSCB  DS    XL9Ø                     SPACE FOR FORMAT 4 DSCB
*
@EDTI    EDTINFO MF=(L,@EDTIN,ØD)
@EDTDATA DS    4D                       PLACE FOR DATA FROM EDTINFO
*
@LSPACE  LSPACE MF=L                    MAP OUT AREA FOR EXECUTION
@LSPACEW DS    XL4Ø
@LSPACED LSPACE MF=(D,DATA)             AREA FOR LSPACE TO PLACE DATA
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  WE USE TWO DIFFERENT DSECTS FOR THE RETURNED CHUNK.  STR1 IS FOR   *
*  THE FIRST ENTRY IN EACH CHUNK.  IT CONTAINS CONTROL INFORMATION    *
*  ABOUT THE CHUNK ITSELF.  STR2 MAPS EACH INDIVIDUAL UCB ENTRY IN    *
*  THE CHUNK.                                                         *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
UCB_STR1 DSECT                          STRUCTURE DEFINITION
UCB_STOR DS    ØXL52                    SPECIFY SIZE
UCB_##   DS    F                        NUMBER OF ENTRIES IN CHUNK
UCB_FP   DS    A                        POINTER TO NEXT CHUNK
UCB_BP   DS    A                        POINTER TO PREVIOUS CHUNK
         DS    (52-(*-UCB_STOR))XL1     FILL IT OUT
UCB_STR2 DSECT                          ANOTHER MAP FOR THE CHUNK
UCB_UNIT DS    4XL1                     4 CHARACTER UCB
UCB_AREA DS    48XL1                    RETURNED COPY OF THE UCB
D_SIZE   EQU   *-UCB_UNIT               LET ASSEMBLER CALCULATE LENGTH
*
UCBDSECT DSECT
         IEFUCBOB DEVCLAS=DA
MYF4     DSECT
         IECSDSL1 (4)
         END   $DSPACE
```

$DSPACE PANEL

```
)ATTR
 | TYPE(OUTPUT)   INTENS(LOW)     COLOR(BLUE)
 ? TYPE(OUTPUT)   INTENS(LOW)     COLOR(GREEN)
 * TYPE(INPUT)    INTENS(LOW)     COLOR(GREEN)
 $ TYPE(TEXT)     INTENS(HIGH)    COLOR(BLUE)
 ~ TYPE(TEXT)     INTENS(HIGH)    COLOR(YELLOW)
 @ TYPE(OUTPUT)   INTENS(HIGH)    COLOR(RED)
 } TYPE(OUTPUT)   INTENS(HIGH)    COLOR(TURQUOISE)
 ! TYPE(TEXT)     INTENS(HIGH)    COLOR(YELLOW) HILITE(REVERSE)
)BODY EXPAND(^^)
```

```
 %^!^<< On-line DASD Device Information >>^+^ +
%Command ===>_ZCMD                                             %Scroll ==
+
           UNIT       MOUNT        VSAM FREE    #      FREE       LARGES
 VOLSER UCB    NAME   STATUS   SMS REF  DSCB   EXT   CYL   TRK   CYL   T
+
)MODEL
}VOL   }UCB }UN      }ST       }SI }VS }DS#  }EXT# }FCYL }FTRK }LCYL }LT
)INIT
)PROC
)END
```

$UCBINFO ROUTINE

```
         TITLE '$UCBINFO - GENERAL UCB SCAN ROUTINE'
*---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+*
* CSECT   : $UCBINFO                                                  *
* MODULE  : $UCBINFO                                                  *
* DESC    : $UCBINFO IS A GENERAL PURPOSE SUBROUTINE THAT CAN BE USED *
*           TO OBTAIN COPIES OF UCBS.  THE OBTAINED INFORMATION IS    *
*           STORED INTO A CHAINED STORAGE STRUCTURE AND PASSED BACK   *
*           TO THE CALLING PROGRAM FOR FURTHER PROCESSING.  THE       *
*           CALLING PROGRAM NEEDS TO SPECIFY THE TYPE OF UCB THAT IS  *
*           DESIRED.                                                  *
* MACROS  : $ESAPRO $ESAEPI $ESASTG STORAGE UCBSCAN                   *
* DSECTS  : UCB_STRU                                                  *
* INPUT   : NONE                                                      *
* OUTPUT  : NONE                                                      *
* PLIST   : STANDARD PARAMETER LIST                                   *
*           PLIST+X'ØØ'  ADDRESS OF 4 BYTE AREA CONTAINING UCB TYPE   *
*           PLIST=X'Ø4'  ADDRESS OF 4 BYTE AREA THAT WILL CONTAIN     *
*                        THE POINTER TO THE STORAGE STRUCTURE THAT    *
*                        CONTAINS THE UCBS                            *
* CALLS   : NONE                                                      *
* NOTES   : NONE                                                      *
*---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+*
         EJECT
$UCBINFO $ESAPRO R12,AM=31,RM=ANY
*
         ST    R1,@PLIST                SAVE PARM POINTER
         L     R2,Ø(R1)                 PICK UP ADDRESS FIRST PARM
         L     RØ,G_SIZE                GET THE SIZE OF AREA TO OBTAIN
*
*---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+*
* OBTAIN FIRST STORAGE AREA FOR UCB INFORMATION                       *
*---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+*
*
         STORAGE OBTAIN,                                               +
               LENGTH=(Ø),                                             +
               LOC=(BELOW,ANY),                                        +
               ADDR=(R11)
*
         ST    R11,@F_POINT            SAVE POINTER FOR LATER
         ST    R11,@R_POINT            SAVE POINTER FOR LATER
         USING UCB_STRU,R11            LET ASSEMBLER KNOW THE BASE
         MVC   CHUNK_S,G_SIZE          SAVE SIZE OF CHUNK IN CHUNK
         LA    R11,D_SIZE(,R11)        INCREMENT ADDRESS
         XR    R1Ø,R1Ø                 CLEAR REGISTER 1Ø
*
         CLC   U_ALL,Ø(R2)             Q. ALL DEVICE TYPES
         BE    UCB_LOOP                A. YES
         CLC   U_CHAR,Ø(R2)            Q. CHARACTER READER DEVICES
         BNE   N_CHAR                  A. NO, NEXT CHECK
         OI    D_CLASS,E_SCAN_XDEVCLASS_CHAR
         B     UCB_LOOP                WE ARE READY TO ROLL
N_CHAR   DS    ØH
         CLC   U_COMM,Ø(R2)            Q. CHARACTER READER DEVICES
         BNE   N_COMM                  A. NO, NEXT CHECK
         OI    D_CLASS,E_SCAN_XDEVCLASS_COMM
         B     UCB_LOOP                WE ARE READY TO ROLL
N_COMM   DS    ØH
         CLC   U_CTC,Ø(R2)             Q. CHARACTER READER DEVICES
         BNE   N_CTC                   A. NO, NEXT CHECK
         OI    D_CLASS,E_SCAN_XDEVCLASS_CTC
         B     UCB_LOOP                WE ARE READY TO ROLL
N_CTC    DS    ØH
         CLC   U_DASD,Ø(R2)            Q. CHARACTER READER DEVICES
         BNE   N_DASD                  A. NO, NEXT CHECK
         OI    D_CLASS,E_SCAN_XDEVCLASS_DASD
         B     UCB_LOOP                WE ARE READY TO ROLL
N_DASD   DS    ØH
         CLC   U_DISP,Ø(R2)            Q. CHARACTER READER DEVICES
         BNE   N_DISP                  A. NO, NEXT CHECK
         OI    D_CLASS,E_SCAN_XDEVCLASS_DISP
         B     UCB_LOOP                WE ARE READY TO ROLL
N_DISP   DS    ØH
         CLC   U_TAPE,Ø(R2)            Q. CHARACTER READER DEVICES
         BNE   N_TAPE                  A. NO, NEXT CHECK
         OI    D_CLASS,E_SCAN_XDEVCLASS_TAPE
         B     UCB_LOOP                WE ARE READY TO ROLL
N_TAPE   DS    ØH
         CLC   U_UREC,Ø(R2)            Q. CHARACTER READER DEVICES
         BNE   N_UREC                  A. NO, NEXT CHECK
         OI    D_CLASS,E_SCAN_XDEVCLASS_UREC
         B     UCB_LOOP                WE ARE READY TO ROLL
*
N_UREC   DS    ØH
*
         MVC   RET_CODE,RCØØ1Ø         SET RETURN TO INDICATE ERROR
         B     EXIT_PGM                EXIT THE PROGRAM
```

```
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  MAIN LOOP FOR OBTAINING THE UCB INFORMATION                        *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
UCB_LOOP DS    ØH
*
         XC    Ø(D_SIZE,R11),Ø(R11)    MAKE SURE THE AREA IS CLEARED
*
         UCBSCAN COPY,                                                 +
               WORKAREA=UCB_WORK,                                      +
               UCBAREA=UCB_AREA,                                       +
               DEVNCHAR=UCB_UNIT,                                      +
               DYNAMIC=YES,                                            +
               RANGE=ALL,                                              +
               DEVCID=D_CLASS,                                         +
               RETCODE=UET_CODE,                                       +
               RSNCODE=USN_CODE,                                       +
               MF=(E,E_SCAN)
*
         CLC   UET_CODE,RCØØØØ         Q. ZERO RETURN CODE
         BNE   EXIT_PGM                A. NO, LOOKS LIKE WE ARE DONE
         LA    R1Ø,1(,R1Ø)             INCREMENT THE COUNTER
         LA    R11,D_SIZE(,R11)        BUMP THE POINTER TO NEXT AREA
         C     R1Ø,F1ØØØ               Q. PROCESSED 1ØØØ ENTRIES
         BNE   UCB_LOOP                A. NO, KEEP GOING
         L     R9,@R_POINT             POINT TO THE BEGINNING OF CHUNK
         ST    R1Ø,Ø(R9)               SAVE THE NUMBER OF ENTRIES
         L     RØ,G_SIZE               GET THE SIZE OF AREA TO OBTAIN
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
*  GET ANOTHER CHUNK OF STORAGE AND CHAIN IT UP TO PREVIOUS CHUNK     *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
         STORAGE OBTAIN,                                               +
               LENGTH=(Ø),                                             +
               LOC=(BELOW,ANY),                                        +
               ADDR=(R11)
*
         MVC   CHUNK_S,G_SIZE          SAVE SIZE OF CHUNK IN CHUNK
         ST    R11,@R_POINT            SAVE THE FORWARD POINTER
         ST    R11,4(R9)               SAVE THE FORWARD POINTER
         ST    R9,8(R11)               STORE THE BACKWARD POINTER
         XR    R1Ø,R1Ø                 CLEAR REG 1Ø
         LA    R11,D_SIZE(,R11)        BUMP POINTER
         B     UCB_LOOP                GO PROCESS THE NEXT UCB
         DROP  R11                     NOTIFY THE ASSEMBLER
*
EXIT_PGM DS    ØH
*
         L     R9,@R_POINT             POINT TO THE BEGINNING OF CHUNK
         ST    R1Ø,Ø(R9)               SAVE THE NUMBER OF ENTRIES
         L     R2,@PLIST               GET THE PLIST POINTER
         L     R2,4(R2)                PICK UP ADDRESS OF SECOND PARM
         MVC   Ø(4,R2),@F_POINT        PASS POINTER BACK TO CALLER
```

```
*
         $ESAEPI RET_CODE
*
NUM_ENT  EQU   1ØØØ                    NUMBER OF ENTRIES IN A BLOCK
G_SIZE   DC    A((D_SIZE*NUM_ENT)+D_SIZE)
F1ØØØ    DC    A(NUM_ENT)
*
U_ALL    DC    CL4'ALL '               ALL DEVICES
U_CHAR   DC    CL4'CHAR'               CHARACTER READER DEVICES
U_COMM   DC    CL4'COMM'               COMMUNICATIONS DEVICES
U_CTC    DC    CL4'CTC '               CHANNEL TO CHANNEL DEVICES
U_DASD   DC    CL4'DASD'               DASD DEVICES
U_DISP   DC    CL4'DISP'               TAPE DEVICES
U_TAPE   DC    CL4'TAPE'               TAPE DEVICES
U_UREC   DC    CL4'UREC'               UNIT RECORD DEVICES
*
         EJECT
         $ESASTG
@PLIST   DS    F                       PLACE TO SAVE PLIST POINTER
@F_POINT DS    F                       POINTER TO STORAGE STRUCTURE
@R_POINT DS    F                       POINTER TO CURRENT CHUNK
RET_CODE DS    F                       PROGRAM RETURN CODE
UET_CODE DS    F                       UCBSCAN RETURN CODE
USN_CODE DS    F                       UCBSCAN REASON CODE
D_CLASS  DS    XL1                     DEVICE TYPE USED BY UCBSCAN
UCB_WORK DS    1ØØXL1                  WORK AREA
         EJECT
         UCBSCAN PLISTVER=MAX,MF=(L,E_SCAN)
         EJECT
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
* STORAGE STRUCTURE WILL USE 52 BYTES PER UCB COPY AND 4 CHARACTER   *
* UCB ADDRESS INFORMATION.  FIRST ENTRY IN THE STRUCTURE IS A SPECIAL *
* LAYOUT, AS DOCUMENTED BELOW.  THE SIZE OF A CHUNCK CAN BE SET BY     *
* THE VALUE OF NUM_ENT.  NUM_ENT IS CURRENTLY 1ØØØ.  YOU CAN SET THIS *
* TO WHATEVER VALUE YOU DESIRE.                                       *
*                                                                     *
*         +---------+                                                 *
*         | 4 BYTES |  NUMBER OF ENTRIES IN A CHUNK                   *
*         +---------+                                                 *
*         | 4 BYTES |  POINTER TO NEXT CHUNK                          *
*         +---------+                                                 *
*         | 4 BYTES |  POINTER TO PREVIOUS CHUNK                      *
*         +---------+                                                 *
*         | 4 BYTES |  SIZE OF THE CHUNK                              *
*         +---------+                                                 *
*         | 36 BYTES|  FILLER                                         *
*         +---------+                                                 *
*         | 4 BYTES |  4 CHARACTER UCB                                *
*         +----+----------------------------+                         *
*         | 48 BYTES                        | COPY OF UCB             *
*         +---------------------------------+                         *
*         | 4 BYTES |  4 CHARACTER UCB                                *
*         +----+----------------------------+                         *
*         | 48 BYTES                        | COPY OF UCB             *
*         +---------------------------------+                         *
*            .                                                        *
*             .     REPEAT                                            *
*              .      UNTIL                                           *
*               .       STRUCTURE                                     *
*                .        IS                                          *
*                 .         FULL                                      *
*         +---------+                                                 *
*         + 4 BYTES |  4 CHARACTER UCB                                *
*         +----+----------------------------+                         *
*         + 48 BYTES                        | COPY OF UCB             *
*         +---------------------------------+                         *
*--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+*
UCB_STRU DSECT                          STRUCTURE DEFINITION
UCB_STOR DS    ØXL52                    SPECIFY SIZE
UCB_##   DS    F                        NUMBER OF ENTRIES IN CHUNK
UCB_FP   DS    A                        POINTER TO NEXT CHUNK
UCB_BP   DS    A                        POINTER TO PREVIOUS CHUNK
CHUNK_S  DS    F                        SIZE OF THE CHUNK
         DS    (52-(*-UCB_STOR))XL1     FILL IT OUT
         ORG   UCB_STOR                 ORG BACK FOR MULTIPLE DEFN.
UCB_UNIT DS    4XL1                     4 CHARACTER UCB
UCB_AREA DS    48XL1                    RETURNED COPY OF THE UCB
D_SIZE   EQU   *-UCB_UNIT               LET ASSEMBLER CALCULATE LENGTH
         END   $UCBINFO
```

$PFPRO MACRO

```
         MACRO
&LABEL   $PFPRO &AM=31,&RM=ANY,&MODE=P
.**********************************************************************
.*       THIS MACRO WILL PROVIDE ENTRY LINKAGE AND OPTIONALLY         *
.*       MULTIPLE BASE REGISTERS.  TO USE THIS MACRO, YOU NEED TO     *
.*       ALSO USE THE $PFSTG MACRO.  THE $PFSTG DEFINES THE SYMBOL    *
.*       QLENGTH WHICH OCCURS IN THE CODE THAT &ESAPRO GENERATES.     *
.*       IF YOU DO NOT CODE ANY OPERANDS, THEN REGISTER 12 WILL BE    *
.*       USED AS THE BASE.  IF YOU CODE MULTIPLE SYMBOLS, THEN THEY   *
.*       WILL BE USED AS THE BASE REGISTERS.                          *
.*                                                                    *
.*       EXAMPLES:                                                    *
.*                                                                    *
.*              SECTNAME $SPFPRO          = REG 12 BASE               *
.*              SECTNAME $SPFPRO 5        = REG 5 BASE                *
.*              SECTNAME $SPFPRO R1Ø,R11  = REGS 1Ø AND 11 ARE BASES  *
.**********************************************************************
*
         LCLA  &AA,&AB,&AC
*
```

```
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
RA       EQU   1Ø
R11      EQU   11
RB       EQU   11
R12      EQU   12
RC       EQU   12
R13      EQU   13
RD       EQU   13
R14      EQU   14
RE       EQU   14
R15      EQU   15
RF       EQU   15
*
&LABEL   CSECT
&LABEL   AMODE &AM
&LABEL   RMODE &RM
*
         SYSSTATE ASCENV=&MODE         SET THE ENVIRONMENT
*
         B     $$$$EYEC-*(R15)         BRANCH AROUND EYECATCHER
         DC    AL1(($$$$EYEC-*)-1)     EYECATCHER LENGTH
$$$$MODI DC    CL8'&LABEL'             MODULE ID
         DC    CL3' - '
$$$$DATE DC    CL8'&SYSDATE'           ASSEMBLY DATE
         DC    CL3' - '
$$$$TIME DC    CL8'&SYSTIME'           ASSEMBLY TIME
         DC    CL3'   '                FILLER
*
$$$$F1SA DC    CL4'F1SA'               USED FOR STACK OPERATIONS
$$$$4Ø96 DC    F'4Ø96'                 USED TO ADJUST BASE REGS
*
$$$$EYEC DS    ØH
*
         BAKR  R14,Ø                   SAVE GPRS AND ARS ON THE STACK
         AIF   (N'&SYSLIST EQ Ø).USER12
         LAE   &SYSLIST(1),Ø(R15,Ø)    LOAD OUR BASE REG
         USING &LABEL,&SYSLIST(1)      LET THE ASSEMBLER KNOW
         AGO   .GNBASE
.USER12  ANOP
         MNOTE *,'NO BASE REG SPECIFIED, REGISTER 12 USED'
         LAE   R12,Ø(R15,Ø)            LOAD OUR BASE REG
         USING &LABEL,R12              LET THE ASSEMBLER KNOW
         AGO   .STGOB
.GNBASE  ANOP
         AIF   (N’&SYSLIST LE 1).STGOB
&AA      SETA  2
&AC      SETA  4Ø96
.GNBASE1 ANOP
*
         AIF   (&AA GT N'&SYSLIST).STGOB
&AB      SETA  &AA-1
         LR    &SYSLIST(&AA),&SYSLIST(&AB) GET INITIAL BASE
         A     &SYSLIST(&AA),$$$$4Ø96      ADJUST NEXT BASE
         USING &LABEL+&AC,&SYSLIST(&AA)    LET THE ASSEMBLER KNOW
&AA      SETA  &AA+1
&AC      SETA  &AC+4Ø96
         AGO   .GNBASE1
.STGOB   ANOP
         L     RØ,QLENGTH              GET THE DSECT LENGTH
         STORAGE OBTAIN,LENGTH=(RØ),LOC=(RES,ANY)
         LR    R15,R1                  GET @(OBTAINED AREA)
         L     R13,QDSECT              GET DISPLACEMENT INTO AREA
         LA    R13,Ø(R13,R15)          GET @(OBTAINED AREA)
         LR    RØ,R13                  SET REG Ø = REG 13
         L     R1,QLENGTH              GET THE LENGTH OF THE AREA
         XR    R15,R15                 CLEAR REG 5
         MVCL  RØ,R14                  INTIALIZE THE AREA
         MVC   4(4,R13),$$$$F1SA       INDICATE STACK USAGE
         USING DSECT,R13               INFORM ASSEMBLER OF BASE
.MEND    ANOP
         LOAD EP=ISPLINK
         ST    RØ,@ISPLINK             SAVE IT
*                                      DEFINE THE STANDARD SPF VARS
         $CALL @ISPLINK,                                               +
               (VDEFINE,STD_VAR,SMSG,STD_TYPE,STD_LEN,LIST),           +
               VL,MF=(E,@CALL)
         EREG  R1,R1                   RESTORE REGISTER 1
         MEND
```

$PFEPI MACRO

```
         MACRO
         $PFEPI
.**********************************************************************
.*       THIS MACRO WILL PROVIDE EXIT LINKAGE. IT WILL FREE THE       *
.*       STORAGE AREA THAT WAS ACQUIRED BY THE $PFPRO MACRO.  YOU     *
.*       CAN OPTIONALLY PASS IT A RETURN CODE VALUE.  THIS VALUE IS   *
.*       EITHER THE LABEL OF A FULL WORD IN STORAGE, OR IT IS A REG-  *
.*       ISTER. AS WITH THE $PFPRO MACRO, YOU NEED TO USE THE $PFSTG  *
.*       MACRO.  THE SYMBOL QLENGTH WHICH OCCURS IN THE CODE THAT IS  *
.*       GENERATED BY THIS MACRO IS DEFINED BY $PFSTG                 *
.*                                                                    *
.*       EXAMPLES:                                                    *
.*                                                                    *
.*              $PFEPI          = NO RETURN CODE SPECIFIED            *
.*              $PFEPI (R5)     = RETURN CODE IS IN REG 5             *
.*              $PFEPI RETCODE  = RETURN CODE IS IN THE FULLWORD AT   *
.*                                RETCODE                             *
.**********************************************************************
*
         $CALL @ISPLINK,                                               +
               (VDELETE,SPF_VAR),                                      +
               VL,MF=(E,@CALL)
*
         AIF   (N'&SYSLIST EQ Ø).STGFRE
*
         AIF   ('&SYSLIST(1)'(1,1) EQ '(').REGRC
         L     R2,&SYSLIST(1)          GET RETURN CODE VALUE
         AGO   .STGFRE
.REGRC   ANOP
         LR    R2,&SYSLIST(1,1)        GET RETURN CODE VALUE
.STGFRE  ANOP
*
         L     RØ,QLENGTH              GET THE DSECT LENGTH
*
         STORAGE RELEASE,LENGTH=(RØ),ADDR=(R13)
*
         AIF   (N'&SYSLIST NE Ø).SETRC
         XR    R15,R15                 CLEAR THE RETURN CODE
         AGO   .MEND
.SETRC   ANOP
         LR    R15,R2                  SET THE RETURN CODE
.MEND    ANOP
         PR                            RETURN TO CALLER
*  FOR ADDRESSABILITY PURPOSES
         LTORG
         MEND
```

$PFSTG MACRO

```
         MACRO
         $PFSTG
.**********************************************************************
.*       THIS MACRO IS USED IN CONJUNCTION WITH THE $PFEPI AND $PFPRO *
.*       MACROS. IT PROVIDES THE DEFINITIONS FOR MANY OF THE VARIABLES*
.*       THAT ARE REQUIRED TO INVOKE VARIOUS ISPF DIALOG SERVICES.  A *
.*       DSECT IS ALSO DEFINED WHICH CONTAINS STORAGE FOR A REGISTER  *
.*       SAVE AREA, AS WELL AS ANY ADDITIONAL STORAGE THE USER MAY    *
.*       DEFINE.  A QCON IS UTILIZED TO ALLOW THE ASSEMBLER TO        *
.*       CALCULATE THE LENGTH OF THE DUMMY SECTION AREA               *
.*                                                                    *
.*       EXAMPLES:                                                    *
.*                                                                    *
.*              $PFSTG                                                 *
.*      XXX     DC    F          = DEFINE ADDITIONAL STORAGE AREA      *
.*      YYY     DC    XL255                                            *
.*       .       .      .                                              *
.*       .       .      .                                              *
.*       .       .      .                                              *
.***********************************************************************
*
* DEFINITIONS FOR ISPF DIALOG OPTIONS
*
*        VARIABLE SERVICES
VGET     DC    CL8'VGET'
VPUT     DC    CL8'VPUT'
VDEFINE  DC    CL8'VDEFINE'
VDELETE  DC    CL8'VDELETE'
VCOPY    DC    CL8'VCOPY'
VREPLACE DC    CL8'VREPLACE'
VRESET   DC    CL8'VRESET'
*        OTHER SERVICES
SELECT   DC    CL8'SELECT'
CONTROL  DC    CL8'CONTROL'
BROWSE   DC    CL8'BROWSE'
EDIT     DC    CL8'EDIT'
EDREC    DC    CL8'EDREC'
LOG      DC    CL8'LOG'
INIT     DC    CL8'INIT'
QUERY    DC    CL8'QUERY'
PROCESS  DC    CL8'PROCESS'
DEFER    DC    CL8'DEFER'
ORDER    DC    CL8'ORDER'
*        FUNCTIONS/MODES
ASIS     DC    CL8'ASIS'
CANCEL   DC    CL8'CANCEL'
DISABLE  DC    CL8'DISABLE'
ENABLE   DC    CL8'ENABLE'
END      DC    CL8'END'
ENTER    DC    CL8'ENTER'
ERRORS   DC    CL8'ERRORS'
LINE     DC    CL8'LINE'
LOCATE   DC    CL8'LOCATE'
LOCK     DC    CL8'LOCK'
MOVE     DC    CL8'MOVE'
NEWCOPY  DC    CL8'NEWCOPY'
NOFT     DC    CL8'NOFT'
NONDISPL DC    CL8'NONDISPL'
NOWRITE  DC    CL8'NOWRITE'
PROFILE  DC    CL8'PROFILE'
REFRESH  DC    CL8'REFRESH'
REPLACE  DC    CL8'REPLACE'
REPLCOPY DC    CL8'REPLCOPY'
RETURN   DC    CL8'RETURN'
RESTORE  DC    CL8'RESTORE'
```

```
SAVE     DC    CL8'SAVE'
SHARED   DC    CL8'SHARED'
SHARE    DC    CL8'SHARE'
SPLIT    DC    CL8'SPLIT'
SM       DC    CL8'SM'
TEMP     DC    CL8'TEMP'
WRITE    DC    CL8'WRITE'
*
* VARIABLE FORMAT TYPES
*
CHAR     DC    CL8'CHAR'
FIXED    DC    CL8'FIXED'
BIT      DC    CL8'BIT'
HEX      DC    CL8'HEX'
BINSTR   DC    CL8'BINSTR'
DBCS     DC    CL8'DBCS'
FLOAT    DC    CL8'FLOAT'
PACK     DC    CL8'PACK'
USER     DC    CL8'USER'
*
* OPTIONS FOR THE VDEFINE SERVICE
*
COPY     DC    CL8'COPY'
NOBSCAN  DC    CL8'NOBSCAN'
LIST     DC    CL8'LIST'
*
* OPTIONS FOR TABLE SERVICES
*
TBCREATE DC    CL8'TBCREATE'
TBADD    DC    CL8'TBADD'
TBTOP    DC    CL8'TBTOP'
TBSORT   DC    CL8'TBSORT'
TBDISPL  DC    CL8'TBDISPL'
TBEND    DC    CL8'TBEND'
*
* OPTIONS FOR MESSAGE SERVICES
*
SETMSG   DC    CL8'SETMSG'
*
* STANDARD VARIABLES USED IN ALL DIALOGS
*
SPF_VAR  DC    C'(*)'                  USED BY THE VDELETE SERVICE
*
STD_VAR  DC    C'(SMSG,LMSG,ALRM,HM,ZCMD,ZEDCMD)'
STD_TYPE DC    6CL8'CHAR'
STD_LEN  DC    F'24',F'72',F’3',F'8',F'6Ø',F'1'
*
QDSECT   DC    Q(DSECT)                DEFINE A QCON
QLENGTH  CXD                           LET ASM CALCULATE THE LENGTH
DSECT    DSECT
         DS    18F                     SET ASIDE REGISTER SAVE AREA
*
```

```
@ISPLINK DS    A                       SET ASIDE SPACE FOR ADDRESS
*
@CALL    DS    2ØF                     SET ASIDE SPACE FOR ADDRESS
*
SMSG     DS    CL24                    SPACE FOR SHORT MESSAGE
LMSG     DS    CL72                    SPACE FOR LONG MESSAGE
ALRM     DS    CL3                     ALARM SETTING
HM       DS    CL8                     HELP MEMU ID
ZCMD     DS    CL6Ø                    COMMAND
ZEDCMD   DS    CL1                     COMMAND
         MEND
```

$ESAPRO MACRO

```
         MACRO
&LABEL   $ESAPRO &AM=31,&RM=ANY,&MODE=P
.**********************************************************************
.*       THIS MACRO WILL PROVIDE ENTRY LINKAGE AND OPTIONALLY
.*       MULTIPLE BASE REGISTERS.  TO USE THIS MACRO, YOU NEED TO
.*       ALSO USE THE $ESASTG MACRO.  THE $ESASTG DEFINES THE SYMBOL
.*       QLENGTH WHICH OCCURS IN THE CODE THAT &ESAPRO GENERATES.
.*       IF YOU DO NOT CODE ANY OPERANDS, THEN REGISTER 12 WILL BE
.*       USED AS THE BASE.  IF YOU CODE MULTIPLE SYMBOLS, THEN THEY
.*       WILL BE USED AS THE BASE REGISTERS.
.*
.*       EXAMPLES:
.*
.*              SECTNAME $ESAPRO          = REG 12 BASE
.*              SECTNAME $ESAPRO 5        = REG 5 BASE
.*              SECTNAME $ESAPRO R1Ø,R11  = REGS 1Ø AND 11 ARE BASES
.**********************************************************************
*
         LCLA  &AA,&AB,&AC
*
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
RA       EQU   1Ø
R11      EQU   11
RB       EQU   11
R12      EQU   12
RC       EQU   12
R13      EQU   13
```

```
RD       EQU   13
R14      EQU   14
RE       EQU   14
R15      EQU   15
RF       EQU   15
*
FPRØ     EQU   Ø
FPR2     EQU   2
FPR4     EQU   4
FPR6     EQU   6
*
&LABEL   CSECT
&LABEL   AMODE &AM
&LABEL   RMODE &RM
         SYSSTATE ASCENV=&MODE         SET THE ENVIRONMENT
         B     $$$$EYEC-*(R15)         BRANCH AROUND EYECATCHER
         DC    AL1(($$$$EYEC-*)-1)     EYECATCHER LENGTH
         DC    CL8'&LABEL'             MODULE ID
         DC    CL3' - '
         DC    CL8'&SYSDATE'           ASSEMBLY DATE
         DC    CL3' - '
         DC    CL8'&SYSTIME'           ASSEMBLY TIME
         DC    CL3'   '                FILLER
*
$$$$F1SA DC    CL4'F1SA'               USED FOR STACK OPERATIONS
$$$$4Ø96 DC    F'4Ø96'                 USED TO ADJUST BASE REGS
*
$$$$EYEC DS    ØH
*
         BAKR  R14,Ø                   SAVE GPRS AND ARS ON THE STACK
         AIF   (N’&SYSLIST EQ Ø).USER12
         LAE   &SYSLIST(1),Ø(R15,Ø)    LOAD OUR BASE REG
         USING &LABEL,&SYSLIST(1)      LET THE ASSEMBLER KNOW
         AGO   .GNBASE
.USER12  ANOP
         MNOTE *,'NO BASE REG SPECIFIED, REGISTER 12 USED'
         LAE   R12,Ø(R15,Ø)            LOAD OUR BASE REG
         USING &LABEL,R12              LET THE ASSEMBLER KNOW
         AGO   .STGOB
.GNBASE  ANOP
         AIF   (N'&SYSLIST LE 1).STGOB
&AA      SETA  2
&AC      SETA  4Ø96
.GNBASE1 ANOP
*
         AIF   (&AA GT N’&SYSLIST).STGOB
&AB      SETA  &AA-1
         LR    &SYSLIST(&AA),&SYSLIST(&AB) GET INITIAL BASE
         A     &SYSLIST(&AA),$$$$4Ø96      ADJUST NEXT BASE
         USING &LABEL+&AC,&SYSLIST(&AA)    LET THE ASSEMBLER KNOW
&AA      SETA  &AA+1
&AC      SETA  &AC+4Ø96
```

```
         AGO   .GNBASE1
.STGOB   ANOP
         L     RØ,QLENGTH                GET THE DSECT LENGTH
         STORAGE OBTAIN,LENGTH=(RØ),LOC=(RES,ANY)
         LR    R15,R1                    GET @(OBTAINED AREA)
         L     R13,QDSECT                GET DISPLACEMENT INTO AREA
         LA    R13,Ø(R13,R15)            GET @(OBTAINED AREA)
         LR    RØ,R13                    SET REG Ø = REG 13
         L     R1,QLENGTH                GET THE LENGTH OF THE AREA
         XR    R15,R15                   CLEAR REG 5
         MVCL  RØ,R14                    INTIALIZE THE AREA
         MVC   4(4,R13),$$$$F1SA         INDICATE STACK USAGE
         USING DSECT,R13                 INFORM ASSEMBLER OF BASE
.MEND    ANOP
         EREG  R1,R1                     RESTORE REGISTER 1
         MEND
```

$ESAEPI MACRO

```
         MACRO
         $ESAEPI
.**********************************************************************
.*
.*       THIS MACRO WILL PROVIDE EXIT LINKAGE. IT WILL FREE THE
.*       STORAGE AREA THAT WAS ACQUIRED BY THE $ESAPRO MACRO.  YOU
.*       CAN OPTIONALLY PASS IT A RETURN CODE VALUE.  THIS VALUE IS
.*       EITHER THE LABEL OF A FULL WORD IN STORAGE, OR IT IS A REG-
.*       ISTER. AS WITH THE $ESAPRO MACRO, YOU NEED TO USE THE $ESASTG
.*       MACRO.  THE SYMBOL QLENGTH WHICH OCCURS IN THE CODE THAT IS
.*       GENERATED BY THIS MACRO IS DEFINED BY $ESASTG
.*
.*       EXAMPLES:
.*
.*              $ESAEPI          = NO RETURN CODE SPECIFIED
.*              $ESAEPI (R5)     = RETURN CODE IS IN REG 5
.*              $ESAEPI RETCODE  = RETURN CODE IS IN THE FULLWORD AT
.*                                 RETCODE
.*
.**********************************************************************
*
         AIF   (N'&SYSLIST EQ Ø).STGFRE
         AIF   ('&SYSLIST(1)'(1,1) EQ '(').REGRC
         L     R2,&SYSLIST(1)            GET RETURN CODE VALUE
         AGO   .STGFRE
.REGRC   ANOP
         LR    R2,&SYSLIST(1,1)          GET RETURN CODE VALUE
.STGFRE  ANOP
         L     RØ,QLENGTH                GET THE DSECT LENGTH
         STORAGE RELEASE,LENGTH=(RØ),ADDR=(R13)
         AIF   (N'&SYSLIST NE Ø).SETRC
         XR    R15,R15                 CLEAR THE RETURN CODE
         AGO   .MEND
.SETRC   ANOP
         LR    R15,R2                  SET THE RETURN CODE
.MEND    ANOP
         PR                            RETURN TO CALLER
* FOR ADDRESSABILITY PURPOSES
         LTORG
         MEND
```

$ESASTG MACRO

```
         MACRO
         $ESASTG
.**********************************************************************
.*
.*       THIS MACRO IS USED IN CONJUNCTION WITH THE $ESAEPI AND $ESAPRO
.*       MACROS.  IT PROVIDES A Q TYPE ADDRESS CONSTANT WHICH WILL CON-
.*       TAIN THE LENGTH OF THE DSECT. A REGISTER SAVE AREA ID PROVIDED
.*       AS WELL.
.*
.*       EXAMPLES:
.*
.*              $ESASTG
.*      XXX     DC    F          = DEFINE ADDITIONAL STORAGE AREA
.*      YYY     DC    XL255
.*       .       .      .
.*       .       .      .
.*       .       .      .
.*
.**********************************************************************
RCØØØØ   DC    F'Ø'                    USED TO SET RETURN CODES
RCØØØ4   DC    F'4'                    USED TO SET RETURN CODES
RCØØØ8   DC    F'8'                    USED TO SET RETURN CODES
RCØØØC   DC    F'12'                   USED TO SET RETURN CODES
RCØØ1Ø   DC    F'16'                   USED TO SET RETURN CODES
QDSECT   DC    Q(DSECT)                DEFINE A QCON
QLENGTH  CXD                           LET ASM CALCULATE THE LENGTH
DSECT    DSECT
         DS    18F                     SET ASIDE REGISTER SAVE AREA
         MEND
```

$CALL MACRO

```
         MACRO
&NAME    $CALL &ENTRY,&OPRNDS,&VLPARA,&BM=BALR,&ID=,&MF=I
.**********************************************************************
.*                                                                    *
.*       MODIFIED VERSION OF THE IBM SUPPLIED CALL MACRO              *
.*                                                                    *
.**********************************************************************
         GBLB  &IHBSWA,&IHBSWB
         GBLC  &IHBNO
         LCLC  &GNAME
&IHBNO   SETC  '3Ø9'
&GNAME   SETC  'IHB'.'&SYSNDX'
&IHBSWA  SETB  ('&VLPARA' EQ 'VL')
&IHBSWB  SETB  ('&ENTRY' EQ '(15))
         AIF   ('&VLPARA' NE '' AND '&VLPARA' NE 'VL').ERROR4
         AIF   ('&MF' EQ 'L' AND '&ENTRY' NE '').ERROR1
         AIF   ('&MF' EQ 'L' AND '&ID' NE '').ERROR2
         AIF   ('&MF' NE 'L' AND '&ENTRY' EQ '').ERROR3
&NAME    DS    ØH                                ALIGNMENT
         AIF   ('&MF' EQ 'L' ).CONTC
         AIF   (&IHBSWB).CONTCC
.CONTC   AIF   ('&OPRNDS' EQ '' AND                                    X
               ('&MF' EQ 'I' OR '&MF' EQ 'L')).CONTB
.CONTA   IHBOPLTX &ENTRY,&OPRNDS,&NAME,MF=&MF
.CONTB   AIF   ('&MF' EQ 'L').EXIT
         AIF   (&IHBSWB).CONTD
         L     15,&ENTRY                         LOAD 15 WITH ENTRY ADR
.CONTD   AIF   ('&BM' EQ 'BASSM').CONTE
         BALR  14,15                             BRANCH TO ENTRY POINT
         AGO   .CONTF
.CONTE   BASSM 14,15                             BRANCH TO ENTRY POINT
.CONTF   AIF   ('&ID' EQ '').EXIT
         DC    X’47ØØ'                           NOP INSTRUCTION WITH
         DC    AL2(&ID)                          ID IN LAST TWO BYTES
.EXIT    MEXIT
.CONTCC  ANOP
&NAME    DS    ØH
         AGO   .CONTC
.ERROR1  IHBERMAC 73,&IHBNO,&ENTRY          ENTRY  W/ MF=L
         MEXIT
.ERROR2  IHBERMAC 74,&IHBNO,&ID             ID W/ MF=L
         MEXIT
.ERROR3  IHBERMAC 26,&IHBNO                 ENTRY SYMBOL MISSING
         MEXIT
.ERROR4  IHBERMAC 1Ø14,THIRD                INVALID THIRD PARM
         MEND
```

Enterprise Data Technologies © Xephon 1998

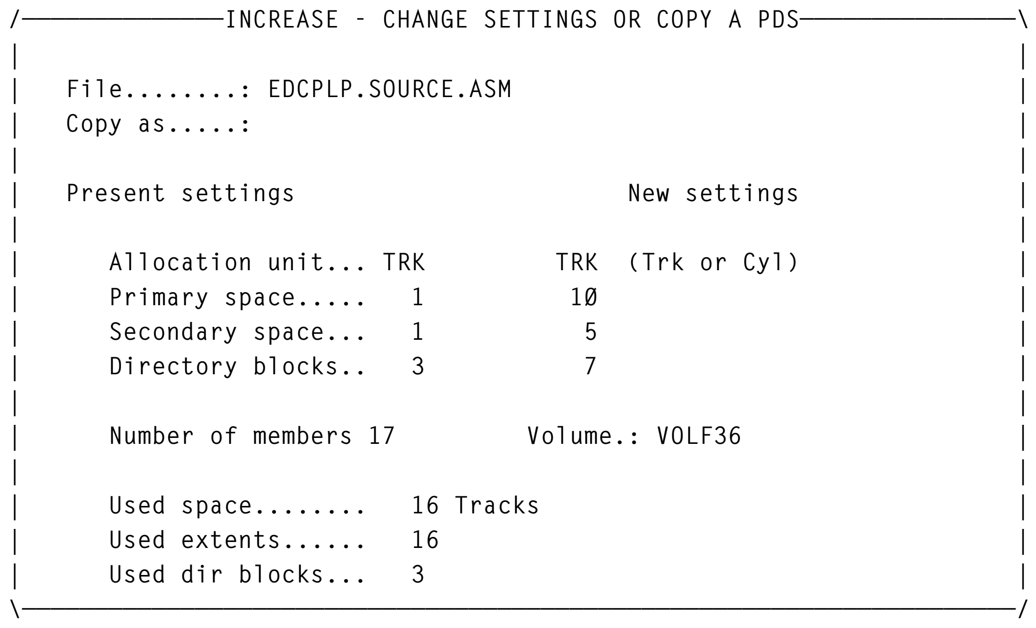
Increasing the space allocation of a PDS

THE PROBLEM

E37-04 is probably a well-known system code for many people, as well as the “No space in directory” message. The traditional approach to the lack of space in a directory is to manually allocate another file with more space, copy everything to it, delete the old one, and rename the new file with the old name.

A SOLUTION

INCREASE is an EXEC to automate these procedures. If you invoke the EXEC passing as an argument the name of your out-of-space PDS, you will get a panel looking like the one shown below in Figure 1.



```
/--------------INCREASE - CHANGE SETTINGS OR COPY A PDS--------------\
|                                                                     |
|   File........: EDCPLP.SOURCE.ASM                                   |
|   Copy as.....:                                                     |
|                                                                     |
|   Present settings                      New settings                |
|                                                                     |
|      Allocation unit... TRK         TRK  (Trk or Cyl)               |
|      Primary space.....   1          1Ø                             |
|      Secondary space...   1           5                             |
|      Directory blocks..   3           7                             |
|                                                                     |
|      Number of members 17         Volume.: VOLF36                   |
|                                                                     |
|      Used space........   16 Tracks                                 |
|      Used extents......   16                                        |
|      Used dir blocks...   3                                         |
\---------------------------------------------------------------------/
```

Figure 1: Example panel

On the left side of the panel, you can see the present settings of the file. On the right side, you can specify the new settings. By default, the new settings have the same values as the old ones, except that any non-track allocation is converted to tracks. For example, if the present settings have one CYL of primary space, in the new settings it will appear as 15 TRK. The same kind of conversion is made for blocks.

Specify the new settings according to your needs – you may just want to increase the directory blocks, increase (or decrease) allocated space, or both. The ‘used’ information provided at the bottom left of the panel can help you decide what to do. You can also change the volume if you wish.

The ‘Copy as’ field at the second line of the panel is optional and can be used to make a copy of the old file with another name, with or without new settings. In this way, the original file will remain as it is. If you leave this field blank, the old file characteristics will be updated to the new ones.

INCREASE creates and submits a job to perform the requested task. The job is prefixed with the user’s name. There are two possible job formats depending on the requested operation.

- If you asked for a ‘Copy as’, the job will be a single-step IEBCOPY.
- If you are changing the original file settings (leaving the above field blank), the job will have three steps:
  - Copy the original to a temporary file allocated with the new settings (this new file has the old one’s name with a .XNEW extension).
  - Rename the old to a backup name (suffixed .XBACK) and rename the .XNEW to the original name.
  - Delete .XBACK.

You should check the job output to see if everything went OK. If something went wrong (for example, you have an overallocated file, and you are downsizing it, and you underspecify the new settings), the job may abort in the first step, leaving a temporary file behind that you must delete by hand before repeating the process (the .XNEW file). If everything is OK, all the steps will have a zero return code.

INCREASE EXEC

```
/*  REXX MVS
/*===================================================================*/
/*                                                                   */
/*  INCREASE - Change allocation characteristics of a PDS.           */
/*             Can also be used to copy it to another PDS.           */
/*             Uses ISPF panel of the same name.                     */
/*                                                                   */
/*===================================================================*/
arg ficheiro .
if ficheiro = "" then do
   say " File? "
   pull ficheiro .
   if ficheiro = "" then exit
end

xx = listdsi(ficheiro directory)     /* Get file characteristics */
if sysreason = 12 then do
   say "INCREASE does not support VSAM files" /* No VSAM, please */
   exit
end
if sysreason¬= 0  then do                  /* Some other mistake */
   say "LISTDSI ReturnCode: " sysreason
   say sysmsglvl1
   say sysmsglvl2
   exit
end
if sysdsorg ¬= "PO" then do                /* File must be a PDS */
   say "File must be a PDS"
   exit
end

 lf1 = strip(ficheiro,,"'")                 /* File name         */
 vol = sysvolume                            /* Volume            */
 au1 = left(sysunits,3)                     /* Allocation unit   */
 pr1 = sysprimary                           /* Primary space     */
 se1 = sysseconds                           /* Secondary space   */
 di1 = sysadirblk                           /* Directory blocks  */
 us1 = sysused                              /* Used space        */
 ux1 = sysextents                           /* Allocated extents */
 ud1 = sysudirblk                           /* Used dir blocks   */
 me1 = sysmembers                           /* Number of members */
 if au1 = 'BLO' then au1 = 'BLK'
 if au1 = 'TRA' then au1 = 'TRK'
 di2 = di1

 if au1 = "CYL" then do                     /* Convert cylinders */
    au2 = "TRK"                             /* to tracks         */
    pr2 = pr1*15
    se2 = se1*15
    us2 = us1*15
    us1 = us1 "Cylinders ("us2" Tracks)"
 end
 if au1 = "BLK" then do                    /* Convert blocks    */
    au2 = "TRK"                            /* to tracks         */
    pr2 = pr1*sysblksize/47ØØØ%1           /* Assume 47k/track  */
    if pr2 < 1 then pr2 = 1
    se2 = se1*sysblksize/47ØØØ%1
    if se2 < 1 then se2 = 1
    us2 = us1*sysblksize/47ØØØ%1
    if us2 < 1 then us2 = 1
    us1 = us1 "Blocks ("us2" Tracks)"
 end
 if au1 = "TRK" then do
    au2 = au1
    pr2 = pr1
    se2 = se1
    us1 = us1 "Tracks"
 end
                                             /* display panel     */
ADDRESS ISPEXEC
'VPUT (LF1 VOL AU1 PR1 SE1 DI1 US1 UX1 UD1 ME1 AU2 PR2 SE2 DI2)'
'ADDPOP ROW(3) COLUMN(2)'
'DISPLAY PANEL(INCREASE)'
if rc=8 then signal saida                    /* rc=8, PF3 or PF15 */
'VGET (LF2 AU2 PR2 SE2 DI2 VOL)'
ADDRESS TSO

call alocar

dropbuf
queue "//"userid()"1 JOB CLASS=A,MSGCLASS=X,REGION=1M"
queue "//*"
queue "//COPYØ EXEC PGM=IEBCOPY"
queue "//SYSPRINT DD SYSOUT=*"
queue "//SYSIN    DD DUMMY"
if lf2="" then do
  queue "//SYSUT1 DD DISP=OLD,DSN="lf1
  queue "//SYSUT2 DD DSN=”lf1".XNEW,"
end
else do
  queue "//SYSUT1 DD DISP=SHR,DSN="lf1
  queue "//SYSUT2 DD DSN="lf2","
end
queue "// DISP=(NEW,CATLG,DELETE),"
queue "// SPACE=("au2",("pr2",”se2","di2")),"
queue "// VOL=SER="vol","
queue "// DCB=(RECFM="sysrecfm",BLKSIZE="sysblksize",LRECL="syslrecl")"
if lf2 = "" then do
```

```
  queue "//*”
  queue "//RENAME EXEC PGM=IKJEFTØ1,COND=(4,LT,COPYØ)”
  queue "//SYSPRINT DD SYSOUT=*"
  queue "//SYSTSPRT DD SYSOUT=*"
  queue "//SYSTSIN  DD *"
  queue " RENAME '"lf1"' '"lf1".XBACK'"
  queue " RENAME "lf1".XNEW' '"lf1"'"
  queue "/*"
  queue "//*"
  queue "//DELETE EXEC PGM=IDCAMS,COND=(4,LT,RENAME)"
  queue "//SYSPRINT DD SYSOUT=*"
  queue "//SYSIN    DD *"
  queue " DELETE '"lf1".XBACK' PURGE"
  queue "/*"
  queue "//"
end
queue ""
"execio * diskw jobe (finis"
"submit '"jobnome"'"
"free dd (jobe)"

saida:
 ADDRESS ISPEXEC ,
'VERASE (LF1 LF2 VOL AU1 PR1 SE1 DI1 US1 UX1 UD1 ME1 AU2 PR2 SE2 DI2)'
 exit

/*================================================================*/
                                      /* Alloc JCL temporary file */
alocar:
 xx = msg(off)
 jobnome = userid()".JOBTEMP"
 "free  dd (jobe)"
 "alloc da('"jobnome"') dd(jobe) new reuse blksize(8ØØØ),
  lrecl(8Ø) recfm(f,b) dsorg(ps) space(1 1) tracks delete "
 if rc™=Ø then do
    say "Error "rc" allocating "jobnome
    exit
 end
return
```

INCREASE PANEL

```
)ATTR
  _ TYPE(INPUT) CAPS(ON) JUST(LEFT)  COLOR(RED)
  # TYPE(INPUT) CAPS(ON) JUST(RIGHT) COLOR(RED)
  ? TYPE(TEXT) INTENS(HIGH) SKIP(ON) COLOR(PINK)
  % TYPE(TEXT) INTENS(HIGH) SKIP(ON) COLOR(YELLOW)
  $ TYPE(TEXT) INTENS(LOW)  SKIP(ON) COLOR(BLUE)
  + TYPE(TEXT) INTENS(LOW)  SKIP(ON) COLOR(GREEN)
  ! TYPE(OUTPUT) CAPS(OFF)  SKIP(ON) COLOR(WHITE)
)BODY WINDOW(7Ø,18)
+
?  File........:_LF1                                         +
?  Copy as.....:_LF2                                         +
+
%  Present settings             New settings
%
+     Allocation unit..!AU1  +    _AU2+ (Trk or Cyl)
+     Primary space....!PR1  +   #PR2 +
+     Secondary space..!SE1  +   #SE2 +
+     Directory blocks.!DI1  +   #DI2 +
+
+     Number of members!ME1  +  %Volume.:_VOL   +
+
+     Used space.......!US1
+     Used extents.....!UX1  +
+     Used dir blocks..!UD1  +
+
+
)INIT
&ZWINTTL = 'INCREASE - Change settings or copy a PDS'
)PROC
&au2ver='TRA TRK CYL'
VER(&AU2,NONBLANK,listv,&au2ver)
VER(&PR2,NONBLANK,num)
VER(&SE2,NONBLANK,num)
VER(&DI2,NONBLANK,num)
VER(&LF1,NONBLANK,dsname)
VER(&LF2,dsname)
VER(&VOL,NONBLANK)
VPUT (LF2 AU2 PR2 SE2 DI2 VOL) SHARED
)END
```

Luis Paulo Figueiredo Sousa Ribeiro
Systems programmer
Edinfor (Portugal)

Migration to a new ISPF release

INTRODUCTION

IBM usually advises about changes to things such as profiles, table structures, or recovery datasets when you migrate to a new ISPF release. However, it assumes that some parts of ISPF are left standard, and that is not always the case. Check the following.

PRIMARY PANEL

The ISPF primary option panel (ISR@PRIM) has had small changes in each of the Version 4 releases. If you have modified this panel (usually to add extra options) you should re-customize the new base panel with each release. Otherwise, something may not work correctly.

In the various ISPF Version 4.*x* releases IBM has gradually increased the number of attribute fields defined in this panel, and sometimes changed the old definitions too. You should not change them.

If you are adding options you should add lines of text in the ')AREA' section of the panel, but be careful to put the same attribute bytes (which are non displaying hex codes) in the same positions as IBM uses for its options, for example:

```
)AREA SAREA39
.N .New option   .This is an option to do something new.
```

The ')PROC' section needs the option name and action, like the following:

```
&ZSEL = TRANS (TRUNC (&ZCMD,'.')
  N,'CMD(%NEWOPT) NEWAPPL'     /* NEWOPT is EXEC for 'New option' */
```

Also, do not forget to add FIELD definitions in the ')PNTS' section when you add options, like the following:

```
FIELD(ZPS01014) VAR(ZCMD) VAL(N)  /* point & shoot for option 'N' */
```

You can also add options to the pull-downs at the top of the screen by adding your definition in one of the)ABC sections, for example:

```
PDC DESC('New option') MNEM(1) ACTION RUN(ISRROUTE)
                       PARM('SELECT CMD(%NEWOPT) NEWAPPL')
```

PRIMARY PANEL OPTIONS AND USER PANELS

Some users use their own ISPF primary option panel, customized to their own requirements. (They have been mostly based on ISPF 3.*x* primary options panels and don’t give the full functionality of the standard ISR@PRIM panel, but that is the user’s choice.) However, when ISPF is upgraded to a new release, there are sometimes changes in the options or on the invocation of options. For example, up until ISPF Version 3.5 the ISPF primary option 0 was “Options” and it selected panel ISPOPTA, but now ISPF Version 4 has “Settings” for option 0 and it selects the program ISPISM (and panel ISPOPTA is no longer supplied).

To make it easier for the migration to Version 4, the following panel was developed for compatibility. It was put in a library which was allocated in every user’s ISPPLIB concatenation; but only users who had their own primary panel with the old option would ever invoke it. Thus the user tries to select the old panel but gets the new program instead.

```
Panel: ISPOPTA for ISPF 4.x

%-------------  ISPF PARAMETER OPTIONS  -------------
%OPTION  ===>_ZCMD
%
%   0 +SETTINGS    - Settings for ISPF 4.x
)INIT
  .RESP = ENTER  /* simulate pressing ENTER without displaying this panel    */
  IF (&SETPANL = &Z)
      &ZCMD = 0  /* to invoke ISPF 4.x SETTINGS program immediately           */
      &SETPANL = YES
  ELSE
      &ZCMD = X  /* to RETURN immediately                                     */
      &SETPANL = &Z
)PROC
  &ZSEL = TRANS( &ZCMD,
                0,'PGM(ISPISM)'          /* do the right thing                */
                X,'EXIT')
/*----------------------------------------------------------------------------*/
/*     This panel would be invoked if the user tries selecting                */
/*     'PANEL(ISPOPTA)' for option 0 instead of 'PGM(ISPISM)'.                */
/*                                                                            */
/*     This panel is never actually seen by the user ( .RESP=ENTER )          */
/*     By this method, the ISPF Version 4 "Settings" is invoked               */
/*     even though the old panel was selected!                                */
/*----------------------------------------------------------------------------*/
)END
```

Now that ISPF 3.5 is out of service, most sites have already upgraded to Version 4, but the technique shown in this panel could be useful in the future too. Otherwise, all you can do is to *advise all users when an option is changed* on your 'official' primary panel and let the users update their own panels appropriately.

The above advice also applies when users maintain 'Group' selection panels to invoke any standard options.

ISPF COMMAND TABLES

The standard ISPF commands are in the ISPCMDS table. It has also been changed from release to release, adding extra functions to ISPF. Therefore, if your site has been modifying this table, you should check the new base ISPCMDS every time you upgrade ISPF. If it has changed you should redo your customization starting with the new table. Better still, all ISPF releases after 4.1 let you define a Site table for your extra commands (in ISPF configuration module ISRCONFG). Use that and leave the ISPCMDS table standard.

Some users have been maintaining their own copy of ISPCMDS so they can use their own special commands. Such users must be warned when a new ISPF will be implemented so they can update their own command tables too. Alternatively, it's better to define a User command table (in ISRCONFG) and advise the users about it.

User tip: if you want to bypass the command table and pass a command directly to your application, type the character '>' before the command on the command line, eg >PRINT to pass 'PRINT' to the application.

I have developed a tool to help with ISPF command customization, and it will be detailed in the next issue.

Ron Brown
Systems Programmer (Germany)

DASD space monitoring

INTRODUCTION

A frequent problem in performance reporting and monitoring is the manipulation and management of the vast amounts of data produced by SMF, RMF, and third-party product reporters. Various data reduction and reporting tools have evolved over the years to address this problem, perhaps one of the most widely installed being Barry Merrill’s SAS/MXG product. The software provides a basic set of SAS routines that reformat raw SMF data into SAS files (databases). Sets of reports and trending macros are also provided.

The following example demonstrates the power and efficiency of SAS in data manipulation and presentation. First we used the IBM utility DCOLLECT (see JOB SASJDIV). Afterwards the job VOLSPAZ read data from the SAS databases created in the first step (SASJDIV) and created a report.

The following code was developed in an MVS/ESA 5.2, SAS 6.096, and SAS/MXG 13.13 environment. Although levels of MXG and MVS are probably irrelevant, some features of SAS Version 6 are used that do not appear in SAS Version 5 (a competent SAS programmer should be able to remove or re-create these features as required). Specific SAS Version 6 attributes are noted in the example.

SASJDIV

```
//SASJDIV JOB COM,'SASDIV',CLASS=W,MSGCLASS=Ø
//*
//* DESCRIPTION: COLLECT IN CPE FOR CFT AND RACF
//*
//DIV    EXEC SAS,REGION=8M,
//       WORK='15Ø,2Ø',
//         OPTIONS='MEMSIZE=16M DMSBATCH BATCH TERMINAL'
//CFT    DD DSN=SAS.SMF.CFT,DISP=SHR
//SMF    DD DSN=SAS.SMF.RAC,DISP=SHR
//REPORT DD DSN=SAS.BERCY.REPORTS,DISP=SHR
//SASLIST DD SYSOUT=Ø
//SYSIN  DD *
  OPTIONS PAGESIZE=6Ø LINESIZE=132 ;
  %CPSTART(MODE=BATCH,
           SYSTEM=MVS,
             ROOT=SAS.SAS6Ø8.CPE.,
             PDB=SAS.BERCY.DIVPDB.,
             DISP=OLD,
             ROOTSERV=,
             SHARE=N/A,
             MXGSRC=('SAS.BERCY.SOURCLIB' 'SAS.MXG.SOURCLIB'),
             MXGLIB=SAS.MXG.FORMATS
            ) ;

  %INCLUDE SOURCLIB(TYPECFT);
  RUN;
  %INCLUDE SOURCLIB(TYPE8ØA);
  RUN;

  %CMPROCES(,
            COLLECTR=GENERIC,
            TOOLNM=SASDS,
            UNIT=DISK,
            GENLIB=WORK
           );

  %CPREDUCE();

                   /****  DAILY REPORTS *****/

   %INCLUDE REPORT(OPTIONS);
   %INCLUDE REPORT(HIER);
   %INCLUDE REPORT(RJCFT);
   %INCLUDE REPORT(RJRACF1);
   %INCLUDE REPORT(RJRACF2);
/*
//*
//* DELETE FILES AFTER PROCESSING
//*
//DELETE   EXEC PGM=IDCAMS,COND=(Ø,NE,DIV.SAS)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DELETE SAS.SMF.CFT
 DELETE SAS.SMF.RAC
/*
//*
```

VOLSPAZ JCL

```
//VOLSPAZ  JOB EXP,'VOLSPAZ',CLASS=W,MSGCLASS=O,MSGLEVEL=(1,1),
//        NOTIFY=DUNAND,USER=SYSOP8,PASSWORD=MANXX
//DELOUT  EXEC PGM=IDCAMS
```

```
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DELETE EXPL69.DISQUE.LIST
 IF MAXCC <= 8 THEN SET MAXCC=0
/*
//VOLSPACE EXEC SAS,REGION=8M,
//       WORK='200,50',
//         OPTIONS='MEMSIZE=16M DMSBATCH BATCH TERMINAL'
//SOURCLIB DD DSN=SAS.BERCY.REPORTS,DISP=SHR
//LIBRARY  DD DSN=SAS.MXG.FORMATS,DISP=SHR
//SASLIST DD DSN=EXPL69.DISQUE.LIST,DISP=(NEW,CATLG,DELETE),
//        UNIT=SYSDA,SPACE=(TRK,(1,1),RLSE),
//        DCB=(RECFM=F,LRECL=133,BLKSIZE=0),MGMTCLAS=DEL32
//SYSIN   DD *

  OPTIONS PAGESIZE=60 LINESIZE=132 ;
  %LET RETCODE=.;

LIBNAME MONTH 'SAS.BERCY.FICPDB.MONTH' DISP=SHR;
LIBNAME DETAIL'SAS.BERCY.FICPDB.DETAIL' DISP=SHR;
   %INCLUDE SOURCLIB(HIER);
   %INCLUDE SOURCLIB(VOLSPAZ);
  RUN;
/*
```

EXAMPLE OUTPUT

```
title "Disk usage";
footnote "List of disk contents";
options linesize=133  pagesize=68;
options nocenter;
proc print data=detail.dcolvol (where=( jour="&hier")) split='*';
  id  dcvolsr;
var dcmangd dcdvtyp dcdvnum dcvlcap dcalloc dcfresp dcperct;
   sum dcvlcap dcalloc dcfresp dcperct ;
  label dcperct = '% Free'
  label dcalloc = 'alloue'
  label dcdvnum = 'Address'
  label dcdvtyp = 'Type'
  label dcfresp = 'Free'
  label dcvlcap = 'Capacity in bytes'
  label dcmangd = 'SMS managed' ;
run;
```

Claude Dunand (France) © Xephon 1998

Processor configuration and IPL information

OVERVIEW

The ISPF dialog described in this article provides the user with the following information:

- Processor configuration details
- CPU information
- CPU vector and cryptographic information
- HSA information
- IPL information
- IEASYSnn IPL information.

The dialog was designed to run under OS/390 Version 1 Release 2 and above. This is because the dialog utilizes a new control block called the Initialization Parameter Area (IPA), which contains valuable information about the parameters used to customize OS/390 during the IPL process. The IPA contains information such as:

- The LOAD parameter (LOADPARM) used at IPL time
- The IPL LOAD parameter dataset name
- The IPL load dataset device number
- The system name (SYSNAME)
- The hardware processor name (HWNAME)
- The logical partition name (LPARNAME)
- All the information contained in the LOAD*xx* member
- The IPL time IEASYSxx parameter value, with symbolics resolved
- Master catalog information.

Figure 1 illustrates how to locate the IPA.

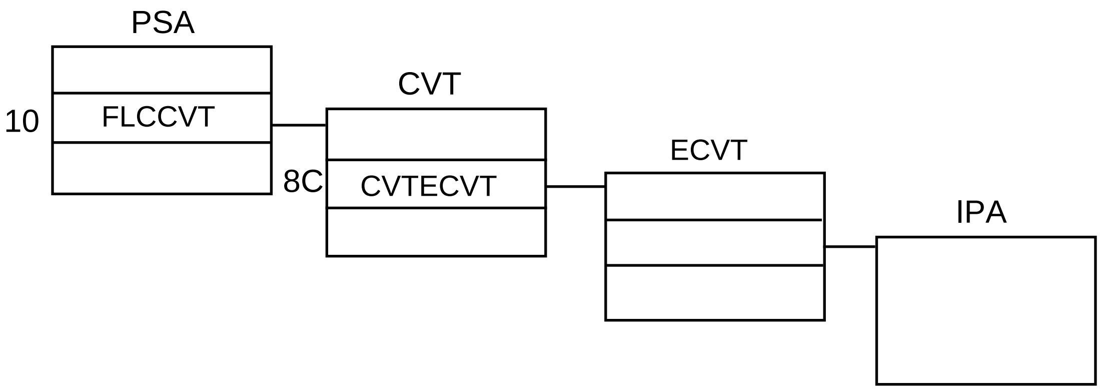


Figure 1: How to locate the IPA

PROCESSOR CONFIGURATION DETAILS

A major part of the information displayed in this section is obtained from the Service Call Control Block (SCCB). The SCCB contains the information returned from the service processor architecture command READ SCP INFO. The SCCB is created at IPL time and is pointed to by the CVTSCPIN field of the CVT. Information displayed includes:

- Real storage configuration
- Extended storage configuration
- Processor information such as:
  - Number of CPUs installed (LPAR mode/non-LPAR mode)
  - Number of HSAs
  - LOADPARM as entered at the service console or HMC
  - Vector information.
- CEC installed facilities, such as whether the CEC has:
  - Initiate reset capability
  - Store status on load
  - Channel path reconfiguration
  - Real storage increment reconfiguration

  - Real storage element reconfiguration
  - Real storage element information
  - Extended storage element reconfiguration
  - Suppress on protection
  - Instruction address buffer
  - Copy and re-assign storage
  - CPU reconfiguration
  - Store channel subsystem characteristics
  - Store status on load
  - Signal alarm.

CPU INFORMATION

For each CPU within the LPAR or non-LPAR configuration, the following information is displayed:

- CPU ID
- CPU address
- PSA virtual address
- PSA real address
- Interrupt subclass mask contained in CR6
- TOD clock number
- Whether the CPU has private space capability
- Whether the CPU has PER2.

HSA INFORMATION

The following information for the HSA(s) is displayed:

- HSA address
- HSA size.

IPL INFORMATION

The IPL information displayed, is broken down into the following:

- OS/390-level information:
  - Product name
  - Product version
  - SP level
  - OS/390 FMID
  - IPL volser
  - IPL date
  - IPL time.
- IPL information:
  - Load parameter (LOADPARM)
  - IODF dataset name
  - Configuration identifier
  - EDT ID
  - LPAR name
  - IEASYSxx suffix
  - IEASYMxx suffix.
- Master catalog information:
  - Master catalog name
  - Master catalog volser
  - Master catalog type
  - Alias name level of qualification
  - CAS service task lower limit.

- IPL LOAD PARMLIB information:
  - PARMLIB dataset name
  - PARMLIB device number.
- Current PARMLIB usage information:
  - PARMLIB dataset name
  - PARMLIB volser
  - Default PARMLIB indicator
  - PARMLIB in use indicator
  - PARMLIB flag settings.

IEASYSxx IPL INFORMATION

This section displays some of the IEASYS*xx* parameters used at IPL time. This includes:

- Page parameter
- PAGTOTL parameter
- VIODSN parameter
- SYSP parameter
- SSN suffix
- SCH suffix
- SCH suffix
- SMF suffix
- LPA suffix
- CLOCK suffix.

CPU VECTOR AND CRYPTO INFORMATION

For each CPU within the LPAR or non-LPAR configuration, the following information is displayed:

- CPU ID
- CPU address
- Whether the vector feature is installed
- Whether the vector feature is connected
- Whether the vector feature is in standby state
- Whether the crypto feature is installed.

DIALOG COMPONENTS

This dialog does not require any special authorization.

SCPINFO

```
         TITLE 'ROUTINE TO EXTRACT CEC INFORMATION'
*-------------------------------------------------------------------*
* NAME:         SCPINFO                                             *
*-------------------------------------------------------------------*
         TITLE 'EQUATES'
RØ       EQU   Ø                       REGISTER Ø
R1       EQU   1                       REGISTER 1
R2       EQU   2                       REGISTER 2
R3       EQU   3                       REGISTER 3
R4       EQU   4                       REGISTER 4
R5       EQU   5                       REGISTER 5
R6       EQU   6                       REGISTER 6
R7       EQU   7                       REGISTER 7
R8       EQU   8                       REGISTER 8
R9       EQU   9                       REGISTER 9
R1Ø      EQU   1Ø                      REGISTER 1Ø
R11      EQU   11                      REGISTER 11
R12      EQU   12                      REGISTER 12
R13      EQU   13                      REGISTER 13
R14      EQU   14                      REGISTER 14
R15      EQU   15                      REGISTER 15
ZERO     EQU   X'ØØ'                   ZERO
SPACE    EQU   C' '                    SPACE
SIGNF    EQU   X'FØ'                   POSITIVE SIGN
SCCBCLEN EQU   X'1Ø'                   SCCB CPU ENTRY LENGTH
SCPINFO  CSECT
SCPINFO  AMODE 31
SCPINFO  RMODE 24
         BAKR  R14,Ø                   SAVE CALLERS ARS + GPRS
*                                      IN THE LINKAGE STACK
         USING SCPINFO,R12             INFORM THE ASSEMBLER
         LAE   R12,Ø(R15,Ø)            SETUP PROGRAM BASE REGISTER
         L     R9,=AL4(WORKALEN)        WORK AREA LENGTH
         STORAGE OBTAIN,LENGTH=(R9),ADDR=(R1Ø),SP=Ø,KEY=8,             X
               LOC=BELOW,COND=NO,RELATED=(FREEWORK,'FREE WORK AREA')
         LAE   R13,Ø(R1Ø,Ø)             @ THE WORKAREA
         USING SAVEAREA,R13             INFORM THE ASSEMBLER
         LA    RØ,SAVEAREA              @ THE WORKAREA
         ICM   R1,B'1111',=AL4(WORKALEN) LENGTH
         SR    R14,R14                  ZEROFILL
         SR    R15,R15                  PROBAGATE
         MVCL  RØ,R14                   CLEAR THE AREA
         MVC   PREVSA,=C'F1SA'          PUT ACRONYM INTO SAVEAREA
*                                       TO INDICATE STATUS SAVED ON
*                                       THE LINKAGE STACK.
         TITLE 'MAIN PROGRAM CONTROL'
CONTROL  EQU   *
         XR    R11,R11                  ZEROISE
         BAS   R2,INIT                  PERFORM INITIALISATION
         BAS   R2,MVSINFOM              EXTRACT THE MVS LEVEL
         LTR   R11,R11                  VARIABLE STORED?
         BNZ   RETURN                   NO-
         BAS   R2,COMNSCP               EXTRACT COMMON SCP AREA
         LTR   R11,R11                  VARIABLE STORED?
         BNZ   RETURN                   NO-
         BAS   R2,CPUINFO               EXTRACT CPU INFO
         LTR   R11,R11                  VARIABLE STORED?
         BNZ   RETURN                   NO-
         BAS   R2,HSAINFO               EXTRACT HSA INFO
         LTR   R11,R11                  VARIABLE STORED?
         BNZ   RETURN                   NO-
         BAS   R2,IPLINFO               EXTRACT IPL INFO
         LTR   R11,R11                  VARIABLE STORED?
         BNZ   RETURN                   NO-
         BAS   R2,IEASYSIF              EXTRACT IEASYSXX NFO
         LTR   R11,R11                  VARIABLE STORED?
         BNZ   RETURN                   NO-
RETURN   EQU   *
         LAE   R1,Ø(R13,Ø)              ADDRESS TO FREE
         L     R9,=AL4(WORKALEN)        WORK AREA LENGTH
         STORAGE RELEASE,ADDR=(R1),LENGTH=(R9),SP=Ø,KEY=8,             X
               COND=NO,RELATED=(GETWORK,'OBTAIN WORK AREA')
EXIT     EQU   *
         LR    R15,R11                  SET RC
         PR                             RESTORE CALLERS AR'S
*                                       GPR'S 2-14 AND RETURN
*                                       TO CALLER
         TITLE 'LETS DO SOME INITIALIZATION'
INIT     EQU   *
*...................................................................
*                                                                  .
* LETS ADDRESS THE CVT AND THE SCCB                                .
*                                                                  .
*...................................................................
         USING PSA,Ø                   INFORM THE ASSEMBLER
         L     R8,CVTPTR               @ OF THE CVT
         SL    R8,=AL4(CVTTCBP-CVTFIX) LENGTH OF THE CVT PREFIX
         USING CVTFIX,R8               INFORM THE ASSEMBLER
         L     R1Ø,CVTSCPIN            @ OF THE SCCB
         USING SCCB,R1Ø                INFORM THE ASSEMBLER
         STCM  R1Ø,B'1111',SCCB@       STORE FOR LATER
         BR    R2                      RETURN TO CALLER
         TITLE 'GET THE MVS LEVEL INFORMATION'
MVSINFOM EQU   *
*...................................................................
*                                                                  .
* GET THE MVS LEVEL INFORMATION                                    .
*                                                                  .
*...................................................................
         STCM  R2,B'1111',RET@         RETURN @
         LA    R7,MVSINFOA             @ DYNAMIC INFO AREA
         USING MVSI,R7                 INFORM THE ASSEMBLER
         MVC   MPRODN,CVTPRODN         SYSTEM PRODUCT LEVEL
         MVC   MPRODI,CVTPRODI         PRODUCT FMID
         ICM   R9,B'1111',CVTECVT      ECVT @
         USING ECVT,R9                 INFORM THE ASSEMBLER
         MVC   MPRODO,ECVTPOWN         PRODUCT OWNER
         MVC   MPRODNM,ECVTPNAM        PRODUCT NAME
         MVC   MPRODVER,ECVTPVER       PRODUCT VERSION
         MVC   MPRODREL,ECVTPREL       PRODUCT RELEASE
         MVC   MPRODMOD,ECVTPMOD       PRODUCT MODIFICATION
         DROP  R9                      INFORM THE ASSEMBLER
         ICM   R9,B'1111',CVTSMCA      SMF SMCA @
         USING SMCABASE,R9             INFORM THE ASSEMBLER
         MVC   MIPLTIME,SMCAITME       IPL TIME
         MVC   MIPLDATE,SMCAIDTE       IPL DATE
         ICM   R9,B'1111',CVTSYSAD     IPL UCB @
         DROP  R9                      INFORM THE ASSEMBLER
         USING UCB,R9                  INFORM THE ASSEMBLER
         MVC   IPLVOL,UCBVOLI          IPL VOLSER
MVCIBLD  EQU   *
         MVC   ECODE,=AL4(TSVEUPDT)    UPDATE OR CREATE A VARIABLE
         LA    R15,MVSLEVEL            @ OF VARIABLE NAME
         STCM  R15,B'1111',PVARPTR     STORE IN PARAMETER LIST
         LA    R15,L'MVSLEVEL(Ø,Ø)     VARIABLE NAME LENGTH
         STCM  R15,B'1111',PVARLEN     STORE IN PARAMETER LIST
         LA    R15,MVSINFOA            @ OF VARIABLE VALUE
         STCM  R15,B'1111',PVARVAL@    STORE IN PARAMETER LIST
         ICM   R5,B'1111',=A(MVSILEN)  LENGTH OF VARIABLE AREA
         STCM  R5,B'1111',PVARVALL     LENGTH OF VARIABLE VALUE
         BAS   R2,IKJCT441             CALL IKJCT441
         ICM   R2,B'1111',RET@         RETURN @
         DROP  R7                      INFORM THE ASSEMBLER
         DROP  R9                      INFORM THE ASSEMBLER
         BR    R2                      RETURN TO CALLER
         TITLE 'EXTRACT THE CPU INFO'
         BR    R2                      RETURN TO CALLER
         TITLE 'GET THE COMMON DEPENDENT AREA'
COMNSCP  EQU   *
```

```
*...................................................................
*                                                                  .
* THIS SUBROUTINE WILL EXTRACT THE COMMON DEPENDENT AREA FROM THE  .
* SCP INFO AREA                                                    .
*                                                                  .
*...................................................................
         STCM  R2,B'1111',RET@         RETURN @
         LA    R7,CSCPAREA             @ DYNAMIC INFO AREA
         USING CSCP,R7                 INFORM THE ASSEMBLER
         MVC   SAR,SCCBSAR             STORAGE ADDRESS RANGE
         MVC   SAI,SCCBSAI             REAL STORAGE ADDRESS INCREMENT
*                                      IN UNITS OF 1M.
         MVC   SBS,SCCBSBS             REAL STORAGE BLOCK SIZE
*                                      IN UNITS OF 1K
         MVC   SII,SCCBSII             REAL STORAGE INCREMENT
*                                      BLOCK INTERLEAVE INTERVAL.
         MVC   NCPS,SCCBNCPS           NUMBER OF CPUS INSTALLED.
         MVC   NHSA,SCCBNHSA           NUMBER OF HSAS.
         MVC   PARM,SCCBPARM           LOAD PARAMETER INFORMATION FROM
         MVC   MESI,SCCBMESI           EXTENDED STORAGE ADDRESS RANGE.
*                                      MAXIMUM EXTENDED STORAGE
*                                      INCREMENT NUMBER INSTALLED.
         MVC   NXSB,SCCBNXSB           NUMBER OF 4K STORAGE BLOCKS IN
*                                      AN EXTENDED STORAGE INCREMENT.
         MVC   MESE,SCCBNXSB           MAXIMUM EXTENDED STORAGE ELEMENT
*                                      NUMBER INSTALLED.
         MVC   VSS,SCCBVSS             VECTOR SECTION SIZE.
         MVC   VPSM,SCCBVPSM           VECTOR PARTIAL SUM NUMBER.
         MVC   IFM1,SCCBIFM1           INSTALLED FACILITY MAP BYTE 1
         MVC   IFM2,SCCBIFM2           INSTALLED FACILITY MAP BYTE 2
         MVC   IFM3,SCCBIFM3           INSTALLED FACILITY MAP BYTE 3
         MVC   IFM4,SCCBIFM4           INSTALLED FACILITY MAP BYTE 4
         MVC   CON1,SCCBCON1           BITS Ø-7 OF CONFIGURATION
*                                      CHARACTERISTICS.
         MVC   CON2,SCCBCON2           BITS 8-15 OF CONFIGURATION
*                                      CHARACTERISTICS
         MVC   ETR,SCCBETR             ETR-SYNC-CHECK TOLERANCE
CSCPBLD  EQU   *
         MVC   ECODE,=AL4(TSVEUPDT)    UPDATE OR CREATE A VARIABLE
         LA    R15,CSCPVAR             @ OF VARIABLE NAME
         STCM  R15,B'1111',PVARPTR     STORE IN PARAMETER LIST
         LA    R15,L'CSCPVAR(Ø,Ø)      VARIABLE NAME LENGTH
         STCM  R15,B'1111',PVARLEN     STORE IN PARAMETER LIST
         LA    R15,CSCPAREA            @ OF VARIABLE VALUE
         STCM  R15,B'1111',PVARVAL@    STORE IN PARAMETER LIST
         ICM   R15,B'1111',=A(CSCPLEN) LENGTH OF VARIABLE AREA
         STCM  R15,B'1111',PVARVALL    LENGTH OF VARIABLE VALUE
         BAS   R2,IKJCT441             CALL IKJCT441
         ICM   R2,B'1111',RET@         RETURN @
         DROP  R7                      INFORM THE ASSEMBLER
         BR    R2                      RETURN TO CALLER
         TITLE 'EXTRACT THE CPU INFO'
CPUINFO  EQU   *
```

```
*...................................................................
*                                                                  .
* THIS SUBROUTINE WILL EXTRACT THE CPU INFO FROM THE SCP INFO AREA .
*                                                                  .
*...................................................................
         STCM  R2,B'1111',RET@         RETURN @
         ICM   R7,B'1111',CVTPCCAT     PCCA VETCTOR TABLE @
         L     R9,Ø(,R7)               PCCA Ø @
         USING PCCA,R9                 INFORM THE ASSEMBLER
         ICM   R4,B'1111',SCCB@        SCCB ADDRESS
         XR    R6,R6                   ZEROISE
         ICM   R6,B'ØØ11',SCCBOCP      OFFSET TO CPU DATA ARRAY
         ALR   R4,R6                   POSITION ONTO THE CPU ARRAY
         USING SCCBCP,R4               INFORM THE ASSEMBLER
         LA    R6,CPUINFOA             OFFSET TO HSA OUTPUT ARRAY
         USING CPUINF,R6               INFORM THE ASSEMBLER
         XR    R5,R5                   ZEROISE
         ICM   R5,B'ØØ11',SCCBNCPS     NUMBER OF CPUS
         C     R5,=AL4(CPUMLEN/CPULEN) MAX ENTIRES?
         BNH   CPUBLD                  NO-
         ICM   R5,B'1111',=AL4(CPUMLEN/CPULEN) MAX ENTIRES?
CPUBLD   EQU   *
         MVC   PCPID,PCCACPID          CPU ID + SERAL NUMBER(PCCACPID)
         MVC   PCPUA,PCCACPUA          CPU ADDRESS(PCCACPUA)
         MVC   PPSAV,PCCAPSAV          VIRTUAL ADDRESS OF PSA(PCCAPSAV)
         MVC   PPSAR,PCCAPSAR          REAL ADDRESS OF PSA(PCCAPSAR)
         MVC   PISCM,PCCAISCM          INTERRUPT SUB-CLASS
*                                      MASK(PCCAISCM)
         MVC   SCPA,SCCBCPA            CPU ADDRESS
         MVC   STOD#,SCCBTOD#          TOD CLOCK NUMBER
         MVC   SCPFL,SCCBCPFL          CPU CHARACTERISTICS FLAG 1
         MVC   SCPF2,SCCBCPF2          CPU CHARACTERISTICS FLAG 2
         LA    R4,SCCBCLEN(,R4)        NEXT ENTRY IN INPUT ARRAY
         LA    R6,CPULEN(,R6)          NEXT ENTRY IN OUTPUT ARRAY
         LA    R7,L'PCCATØØP(,R7)      NEXT PCCA
         L     R9,Ø(,R7)               LET'S ADDRESS THE NEXT PCCA
         BCT   R5,CPUBLD               DO WHILE R5 > Ø?
         XR    R14,R14                 ZEROISE
         XR    R15,R15                 ZEROISE
         ICM   R15,B'ØØ11',SCCBNCPS    NUMBER OF CPS
         M     R14,=AL4(CPULEN)        OUTPUT ARRAY SIZE
         LR    R5,R15                  LENGTH OF VARIABLE AREA
CPUVARV  EQU   *
         MVC   ECODE,=AL4(TSVEUPDT)    UPDATE OR CREATE A VARIABLE
         LA    R15,CPUVAR              @ OF VARIABLE NAME
         STCM  R15,B'1111',PVARPTR     STORE IN PARAMETER LIST
         LA    R15,L'CPUVAR(Ø,Ø)       VARIABLE NAME LENGTH
         STCM  R15,B'1111',PVARLEN     STORE IN PARAMETER LIST
         LA    R15,CPUINFOA            @ OF VARIABLE VALUE
         STCM  R15,B'1111',PVARVAL@    STORE IN PARAMETER LIST
         STCM  R5,B'1111',PVARVALL     LENGTH OF VARIABLE VALUE
         BAS   R2,IKJCT441             CALL IKJCT441
         ICM   R2,B'1111',RET@         RETURN @
         DROP  R4                      INFORM THE ASSEMBLER
         DROP  R6                       INFORM THE ASSEMBLER
         DROP  R9                       INFORM THE ASSEMBLER
         BR    R2                       RETURN TO CALLER
         TITLE 'EXTRACT THE HSA INFO'
HSAINFO  EQU   *
*.....................................................................
*                                                                    .
* THIS SUBROUTINE WILL EXTRACT THE HSA INFO FROM THE SCP INFO AREA .
*                                                                    .
*.....................................................................
         STCM  R2,B'1111',RET@          RETURN @
         ICM   R4,B'1111',SCCB@         SCCB ADDRESS
         XR    R6,R6                    ZEROISE
         ICM   R6,B'ØØ11',SCCBOHSA      OFFSET TO HSA DATA ARRAY
         ALR   R4,R6                    POSITION ONTO THE HSA ARRAY
         USING SCCBHSA,R4               INFORM THE ASSEMBLER
         LA    R6,HSAINFOA              OFFSET TO HSA OUTPUT ARRAY
         USING HSAI,R6                  INFORM THE ASSEMBLER
         XR    R5,R5                    ZEROISE
         ICM   R5,B'ØØ11',SCCBNHSA      NUMBER OF HSAS
         LTR   R5,R5                    CAN THE HSA BE SEEN?
         BNZ   CALCHSAS                 YES-
         XC    HSAZ,HSAZ                ZEROIES
         XC    HSAA,HSAA                ZEROISE
         XR    R14,R14                  ZEROISE
         LA    R15,1(Ø,Ø)               LET'S SAY 1 ENTRY
         M     R14,=AL4(HSALEN)         OUTPUT ARRAY SIZE
         LR    R5,R15                   LENGTH OF VARIABLE AREA
         B     HSAVARV                  BUILD REXX VAR
CALCHSAS EQU   *
         C     R5,=AL4(HSAMLEN/HSALEN) MAX ENTRIES?
         BNH   HSABLD                   NO-
         ICM   R5,B'1111',=AL4(HSAMLEN/HSALEN) MAX ENTRIES?
HSABLD   EQU   *
         MVC   HSAZ,SCCBHSSZ            HSA SIZE
         MVC   HSAA,SCCBAHSA            HSA ADDRESS
         LA    R4,HSALEN(,R4)           NEXT ENTRY IN INPUT ARRAY
         LA    R6,HSALEN(,R6)           NEXT ENTRY IN OUTPUT ARRAY
         BCT   R5,HSABLD                DO WHILE R5 > Ø?
         XR    R14,R14                  ZEROISE
         XR    R15,R15                  ZEROISE
         ICM   R15,B'ØØ11',SCCBNHSA     NUMBER OF HSAS
         M     R14,=AL4(HSALEN)         OUTPUT ARRAY SIZE
         LR    R5,R15                   LENGTH OF VARIABLE AREA
HSAVARV  EQU   *
         MVC   ECODE,=AL4(TSVEUPDT)     UPDATE OR CREATE A VARIABLE
         LA    R15,HSAVAR               @ OF VARIABLE NAME
         STCM  R15,B'1111',PVARPTR      STORE IN PARAMETER LIST
         LA    R15,L'HSAVAR(Ø,Ø)        VARIABLE NAME LENGTH
         STCM  R15,B'1111',PVARLEN      STORE IN PARAMETER LIST
         LA    R15,HSAINFOA             @ OF VARIABLE VALUE
         STCM  R15,B'1111',PVARVAL@     STORE IN PARAMETER LIST
         STCM  R5,B'1111',PVARVALL      LENGTH OF VARIABLE VALUE
         BAS   R2,IKJCT441              CALL IKJCT441
         ICM   R2,B'1111',RET@          RETURN @
         DROP  R4                       INFORM THE ASSEMBLER
         DROP  R6                       INFORM THE ASSEMBLER
         BR    R2                       RETURN TO CALLER
IPLINFO  EQU   *
*.....................................................................
*                                                                    .
* THIS SUBROUTINE WILL EXTRACT THE IPL INFORMATION FROM THE IPA      .
*                                                                    .
*.....................................................................
         STCM  R2,B'1111',RET@          RETURN @
         ICM   R9,B'1111',CVTECVT       ECVT @
         USING ECVT,R9                  INFORM THE ASSEMBLER
         ICM   R9,B'1111',ECVTIPA       IPA @
         USING IPA,R9                   INFORM THE ASSEMBLER
         LA    R6,IPLINFOA              OFFSET TO IPL OUTPUT ARRAY
         USING IPLINF,R6                INFORM THE ASSEMBLER
IPLBLD   EQU   *
         MVC   LPARM,IPALPARM           LOAD PARM
         MVC   IODFHLQ,IPAIOHLQ         IODF HLQ
         MVC   IODFSUF,IPAIOSUF         IODF SUFFIX
         MVC   IOCFG,IPAIOCFG           IO CONFIGURATION ID
         MVC   IOEDT,IPAIOEDT           IO EDT
         MVC   IODDS,IPAIODDS           DEVICE SUPPORT MODULES
         MVC   NUCLSTID,IPANLID         NUCLST MEMBER ID
         MVC   SYSPARM,IPASPLST         LIST OF IEASYS SUFFIXES
         MVC   IEASYM,IPASYLST          LIST OF IEASYM SUFFIXES
         MVC   SYSPLEX,IPASXNAM         SYSPLEX NAME
         MVC   LPARNAME,IPALPNAM        LPAR NAME
         MVC   HWNAME,IPAHWNAM          HARDWARE NAME
         MVC   MCATD,IPASCDSN           MASTER CATALOG DSNAME
         MVC   MCATV,IPASCVOL           MASTER CATALOG VOLUME
         MVC   MCATT,IPASCTYP           MASTER CATALOG TYPE
         MVC   MCATAL,IPASCANL          ALIAS NAME LEVEL
         MVC   CCAS,IPASCCAS            CAS SERVICE TASK LOWER LIMIT
         MVC   PLIBDSN,IPALPDSN         IPL PARMLIB DSN
         MVC   PLIBDDV,IPALPDDV         IPL PARMLIB DEVICE NO
         MVC   PARMDSN,IPAPLDSN         PARMLIB DSN
         MVC   PARMVOL,IPAPLVOL         PARMLIB VOLSER
         MVC   PARMFLAG,IPAPLFLG        PARMLIB USAGE FLAG
IPLVARV  EQU   *
         MVC   ECODE,=AL4(TSVEUPDT)     UPDATE OR CREATE A VARIABLE
         LA    R15,IPLVAR               @ OF VARIABLE NAME
         STCM  R15,B'1111',PVARPTR      STORE IN PARAMETER LIST
         LA    R15,L'IPLVAR(Ø,Ø)        VARIABLE NAME LENGTH
         STCM  R15,B'1111',PVARLEN      STORE IN PARAMETER LIST
         LA    R15,IPLINFOA             @ OF VARIABLE VALUE
         STCM  R15,B'1111',PVARVAL@     STORE IN PARAMETER LIST
         ICM   R15,B'1111',=A(IPLLEN)   LENGTH OF VARIABLE AREA
         STCM  R15,B'1111',PVARVALL     LENGTH OF VARIABLE VALUE
         BAS   R2,IKJCT441              CALL IKJCT441
         ICM   R2,B'1111',RET@          RETURN @
         DROP  R6                      INFORM THE ASSEMBLER
         DROP  R9                      INFORM THE ASSEMBLER
         BR    R2                      RETURN TO CALLER
         TITLE 'IEASYSXX MEMBER INFORMATION'
IEASYSIF EQU   *
         STCM  R2,B'1111',RET@         RETURN @
         ICM   R9,B'1111',CVTECVT      ECVT @
         USING ECVT,R9                 INFORM THE ASSEMBLER
         ICM   R9,B'1111',ECVTIPA      IPA @
         USING IPA,R9                  INFORM THE ASSEMBLER
         LA    R6,SYSINFOA             OFFSET TO IEASYS MEMBER
*                                      DATA ARRAY
         USING IEASYSXX,R6             INFORM THE ASSEMBLER
         USING IPAPDE,R5               INFORM THE ASSEMBLER
CMD      EQU   *
         LA    R5,IPACMD               PDE FOR IEACMDXX?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    PAGEP                   NO-
         LA    R3,CMDVAR               CMD REXX VARIABLE @
         LA    R15,L'CMDVAR(Ø,Ø)       LENGTH OF CMD REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
PAGEP    EQU   *
         LA    R5,IPAPAGEP             PDE FOR PAGE IEASYSXX
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    PAGEO                   NO-
         LA    R3,PAGVARP              PAGE REXX VARIABLE @
         LA    R15,L'PAGVARP(Ø,Ø)      LENGTH OF PAGE REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
PAGEO    EQU   *
         LA    R5,IPAPAGEO             PDE FOR PAGE OP OVERRIDE
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    CON                     NO-
         LA    R3,PAGVARO              PAGE REXX VARIABLE @
         LA    R15,L'PAGVARO(Ø,Ø)      LENGTH OF PAGE REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
CON      EQU   *
         LA    R5,IPACON               PDE FOR CONSOLXX?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    CLOCK                   NO-
         LA    R3,CONVAR               CON REXX VARIABLE @
         LA    R15,L'CONVAR(Ø,Ø)       LENGTH OF CON REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                  NO-
CLOCK    EQU   *
         LA    R5,IPACLOCK              PDE FOR CLOCKXX?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    CSA                      NO-
         LA    R3,CLKVAR                CLOCK REXX VARIABLE @
         LA    R15,L'CLKVAR(Ø,Ø)        LENGTH OF CLOCK REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
SQA      EQU   *
         LA    R5,IPASQA                PDE FOR SQA?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    CSA                      NO-
         LA    R3,SQAVAR                SQA REXX VARIABLE @
         LA    R15,L'SQAVAR(Ø,Ø)        LENGTH OF SQA REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
CSA      EQU   *
         LA    R5,IPACSA                PDE FOR CSA?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    SYSP                     NO-
         LA    R3,CSAVAR                CSA REXX VARIABLE @
         LA    R15,L'CSAVAR(Ø,Ø)        LENGTH OF CSA REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
SYSP     EQU   *
         LA    R5,IPASYSP               PDE FOR SYSP?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    SCH                      NO-
         LA    R3,SYSPVAR               SYSP REXX VARIABLE @
         LA    R15,L'SYSPVAR(Ø,Ø)       LENGTH OF SYSP REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
SCH      EQU   *
         LA    R5,IPASCH                PDE FOR SCH?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    SMF                      NO-
         LA    R3,SCHVAR                SCH REXX VARIABLE @
         LA    R15,L'SCHVAR(Ø,Ø)        LENGTH OF SCH REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
SMF      EQU   *
         LA    R5,IPASMF                PDE FOR SMF?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    SSN                     NO-
         LA    R3,SMFVAR               SMF REXX VARIABLE @
         LA    R15,L'SMFVAR(Ø,Ø)       LENGTH OF SMF REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
SSN      EQU   *
         LA    R5,IPASSN               PDE FOR SSN?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    SVC                     NO-
         LA    R3,SSNVAR               SSN REXX VARIABLE @
         LA    R15,L'SSNVAR(Ø,Ø)       LENGTH OF SSN REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
SVC      EQU   *
         LA    R5,IPASVC               PDE FOR SVC?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    PROG                    NO-
         LA    R3,SVCVAR               SVC REXX VARIABLE @
         LA    R15,L'SVCVAR(Ø,Ø)       LENGTH OF SVC REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
PROG     EQU   *
         LA    R5,IPAPROG              PDE FOR PROG?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    PAGTO                   NO-
         LA    R3,PROGVAR              PROG REXX VARIABLE @
         LA    R15,L'PROGVAR(Ø,Ø)      LENGTH OF PROG REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
PAGTO    EQU   *
         LA    R5,IPAPAGTO             PDE FOR PAGTOTL?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    VIODSN                  NO-
         LA    R3,PAGTVAR              PAGTOTL REXX VARIABLE @
         LA    R15,L'PAGTVAR(Ø,Ø)      LENGTH OF PAGTOTL REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
VIODSN   EQU   *
         LA    R5,IPAVIODS             PDE FOR VIODSN?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    LOGREC                  NO-
         LA    R3,VIODVAR              VIODSN REXX VARIABLE @
         LA    R15,L'VIODVAR(Ø,Ø)       LENGTH OF VIODSN REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
LOGREC   EQU   *
         LA    R5,IPALOGRE              PDE FOR LOGREC?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    LNK                      NO-
         LA    R3,LOGRVAR               LOGREC REXX VARIABLE @
         LA    R15,L'LOGRVAR(Ø,Ø)       LENGTH OF LOGREC REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
LNK      EQU   *
         LA    R5,IPALNK                PDE FOR LNK?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    LPA                      NO-
         LA    R3,LNKVAR                LNK REXX VARIABLE @
         LA    R15,L'LNKVAR(Ø,Ø)        LENGTH OF LNK REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
LPA      EQU   *
         LA    R5,IPALPA                PDE FOR LPA?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    MLPA                     NO-
         LA    R3,LPAVAR                LPA REXX VARIABLE @
         LA    R15,L'LPAVAR(Ø,Ø)        LENGTH OF LPA REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
MLPA     EQU   *
         LA    R5,IPAMLPA               PDE FOR MLPA?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    MSTRJCL                  NO-
         LA    R3,MLPAVAR               MLPA REXX VARIABLE @
         LA    R15,L'MLPAVAR(Ø,Ø)       LENGTH OF MLPA REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                  ALL OK?
         BNZ   ALLDONE                  NO-
MSTRJCL  EQU   *
         LA    R5,IPAMSTRJ              PDE FOR MSTRJCL?
         CLC   IPAPDESA,=AL4(Ø)         PRESENT?
         BE    LNKAUTH                  NO-
         LA    R3,MSTRVAR               MSTRJCL REXX VARIABLE @
         LA    R15,L'MSTRVAR(Ø,Ø)       LENGTH OF MSTRJCL REXX VARIABLE
         STCM  R15,B'1111',VARLEN       STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD               BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
LNKAUTH  EQU   *
         LA    R5,IPALNKAU             PDE FOR LNKAUTH?
         CLC   IPAPDESA,=AL4(Ø)        PRESENT?
         BE    ALLDONE                 NO-
         LA    R3,LNKAVAR              LNKAUTH REXX VARIABLE @
         LA    R15,L'LNKAVAR(Ø,Ø)      LENGTH OF LNKAUTH REXX VARIABLE
         STCM  R15,B'1111',VARLEN      STORE FOR FUTURE REFERENCE
         BAS   R2,PARMBLD              BUILD THE PARAMETER INFORMATION
         LTR   R11,R11                 ALL OK?
         BNZ   ALLDONE                 NO-
ALLDONE  EQU   *
         ICM   R2,B'1111',RET@         RETURN @
         BR    R2                      RETURN TO CALLER
         TITLE 'BUILD THE IEASYSNN PARAMETER INFORMATION'
PARMBLD  EQU   *
         STCM  R2,B'1111',RET2@        RETURN @
         MVC   SOURCE,IPAPDEDO         SOURCE OF PARAMETER STRING
         XR    R1,R1                   ZEROISE
         XR    R7,R7                   ZEROISE
         XR    R15,R15                 ZEROISE
         ICM   R15,B'ØØ11',IPAPDESL    ANY PARAMETER INFO?
         BZ    STORLEN                 NO-
         LA    RØ,PARMSTRG             PARAMETER STRING
         ICM   R1,B'ØØ11',=AL2(L'PARMSTRG) MAX LENGTH TO MOVE
         ICM   R14,B'1111',IPAPDESA    @ OF PDE DESCRIPTION
         CR    R15,R1                  LESS THAN MAX ALLOWED?
         BNL   SETPLEN                 = OR >?
         LR    R1,R15                  SWITCH AROUND
         B     STORLEN                 STORE THE LENGTH
SETPLEN  EQU   *
         LR    R15,R1                  SWITCH AROUND
STORLEN  EQU   *
         LR    R7,R1                   PARM LENGTH
         STCM  R7,B'ØØ11',PARMLEN      AND STORE
         LA    R7,L'PARMLEN+L'SOURCE(,R7) VARIABLE LENGTHS
         MVCL  RØ,R14                  MOVE THE PARAMETER INFO
SYSVARV  EQU   *
         MVC   ECODE,=AL4(TSVEUPDT)    UPDATE OR CREATE A VARIABLE
         STCM  R3,B'1111',PVARPTR      STORE IN PARAMETER LIST
         ICM   R15,B'1111',VARLEN      VARIABLE LENGTH
         STCM  R15,B'1111',PVARLEN     STORE IN PARAMETER LIST
         LA    R15,SYSINFOA            @ OF VARIABLE VALUE
         STCM  R15,B'1111',PVARVAL@    STORE IN PARAMETER LIST
         STCM  R7,B'1111',PVARVALL     LENGTH OF VARIABLE VALUE
         BAS   R2,IKJCT441             CALL IKJCT441
         ICM   R2,B'1111',RET2@        RETURN @
         BR    R2                      RETURN TO CALLER
         DROP  R5                      INFORM THE ASSEMBLER
         DROP  R6                      INFORM THE ASSEMBLER
         DROP  R9                      INFORM THE ASSEMBLER
         TITLE 'CREATE THE REXX VARIABLES'
```

```
IKJCT441 EQU   *
         XC    IKJTOKEN,IKJTOKEN       NO REQUIRED FOR THIS CALL
         XC    RCODE441,RCODE441       RETURN CODE
         L     R15,CVTTVT              TSCT @
         USING TSVT,R15                INFORM THE ASSEMBLER
         L     R15,TSVTVACC            IKJCT441 @
         LTR   R15,R15                 ENTRY POINT FOUND?
         BNZ   CALL441                 YES- DO A CALL
LINK441  EQU   *
         MVC   LINKAREA,LINKL          LINK SL=L
         MVC   CALLAREA,CALLL          PROG PROGRAM LIST
         LINK  EP=IKJCT441,                                            X
               PARAM=(ECODE,           ENTRY CODE                      X
               PVARPTR,                POINTER TO PANEL VAR NAME       X
               PVARLEN,                LENGTH  OF PANEL VAR NAME       X
               PVARVAL@,               POINTER TO PAN VAR VALUE        X
               PVARVALL,               LENGTH  OF PAN VAR VALUE        X
               IKJTOKEN,               TOKEN                           X
               ECTPARM,                NOT REQUIRED                    X
               RCODE441),              RETURN CODE                     X
               VL=1,                   EOL                             X
               MF=(E,CALLAREA),        CALL AREA                       X
               SF=(E,LINKAREA)         LINK AREA
         B     CHKRET                  CHECK THE RETURN CODE
CALL441  EQU   *
         MVC   CALLAREA,CALLL          PROG PROGRAM LIST
         CALL  (15),                                                   X
               (ECODE,                 ENTRY CODE                      X
               PVARPTR,                POINTER TO PANEL VAR NAME       X
               PVARLEN,                LENGTH  OF PANEL VAR NAME       X
               PVARVAL@,               POINTER TO PAN VAR VALUE        X
               PVARVALL,               LENGTH  OF PAN VAR VALUE        X
               IKJTOKEN,               TOKEN                           X
               ECTPARM,                NOT REQUIRED                    X
               RCODE441),              RETURN CODE                     X
               VL,                     EOL                             X
               MF=(E,CALLAREA)
         B     CHKRET                  CHECK THE RETURN CODE
CHKRET   EQU   *
         LR    R11,R15                 RETURN CODE
         BR    R2                      RETURN TO CALLER
         TITLE 'LITERALS'
         LTORG
         TITLE 'STORAGE ITEMS'
CSCPVAR  DC    CL4'CSCP'               CSCP AREA VARIABLE
MVSLEVEL DC    CL8'MVSLEVEL'           MVS LEVEL VARIABLE
HSAVAR   DC    CL4'HSAV'               HSA VARIABLE
CPUVAR   DC    CL4'CPUV'               CPU VARIABLE
IPLVAR   DC    CL4'IPLV'               IPL VARIABLE
CMDVAR   DC    CL4'VCMD'               IEACMDXX VARIABLE
PAGVARP  DC    CL5'VPAGP'              PAGE IEASYSNN
PAGVARO  DC    CL5'VPAGO'              PAGE OPERATOR OVERRIDE
CONVAR   DC    CL4'VCON'               CONSOLXX VARIABLE
```

```
CLKVAR   DC    CL4'VCLK'               CLOCKXX VARIABLE
CSAVAR   DC    CL4'VCSA'               CSA VARIABLE
SQAVAR   DC    CL4'VSQA'               SQA VARIABLE
SYSPVAR  DC    CL5'VSYSP'              SYSP VARIABLE
SCHVAR   DC    CL4'VSCH'               SCH  VARIABLE
SMFVAR   DC    CL4'VSMF'               SMF  VARIABLE
SSNVAR   DC    CL4'VSSN'               SSN  VARIABLE
SVCVAR   DC    CL4'VSVC'               SVC  VARIABLE
LNKVAR   DC    CL4'VLNK'               LNK  VARIABLE
LPAVAR   DC    CL4'VLPA'               LPA  VARIABLE
MLPAVAR  DC    CL5'VMLPA'              MLPA VARIABLE
MSTRVAR  DC    CL5'VMSTR'              MSTJCLXX VARIABLE
PROGVAR  DC    CL5'VPROG'              PROGXX   VARIABLE
PAGTVAR  DC    CL5'VPAGT'              PAGTOTL  VARIABLE
VIODVAR  DC    CL5'VVIOD'              VIODSN   VARIABLE
LOGRVAR  DC    CL5'VLOGR'              LOGREC   VARIABLE
LNKAVAR  DC    CL5'VLNKA'              LNKAUTH  VARIABLE
ECTPARM  DC    X'FFFFFFFF'             ECT
         TITLE 'MACRO LIST AREA'
LINKL    LINK SF=L
LINKLEN  EQU   *-LINKL                 LENGTH
CALLL    CALL ,(,,,,,,,),MF=L
CALLLEN  EQU   *-CALLL                 LENGTH
         TITLE 'WORKAREA DSECT'
WORKAREA DSECT
SAVEAREA DS    CL72                    SAVEAREA
PREVSA   EQU   SAVEAREA+4,4            @ OF PREVIOUS SAVEAREA
RACFVERS DS    X                       RACF VERSION FLAG
         DS    ØF                      ALIGNMENT
RET@     DS    F                       RETURN @
RET2@    DS    F                       RETURN @
RETCODE  DS    F                       RETURN CODE
VARLEN   DS    F                       VARIABLE LENGTH
SCCB@    DS    F                       SCCB @
NOOFFWS  DS    F                       NO OF FULLWORDS
CLASMSK@ DS    F                       RACF CLASS MASK @
DW       DS    D                       WORK AREA
ECODE    DC    AL4(TSVNOIMP)           CREATE CODE
PARMLIST DS    ØF
PVARPTR  DS    F                       VAR PTR
PVARLEN  DS    F                       VAR LEN
PVARVAL@ DS    F                       VAR VALUE @
PVARVALL DS    F                       VAR VAL LENGTH
IKJTOKEN DS    F                       TOKEN
RCODE441 DS    F                       RETURN CODE
         DS    ØF
LINKAREA DS    CL(LINKLEN)             LINK AREA
CALLAREA DS    CL(CALLLEN)             PARM LIST AREA
CSCPAREA DS    CL(CSCPLEN)             COMMON SCP ENTRY AREA
MVSINFOA DS    CL(MVSILEN)             MVS INFORMATION ENTRY AREA
CPUINFOA DS    CL(CPUMLEN)             CPU INFORMATION ENTRY AREA
HSAINFOA DS    CL(HSAMLEN)             HSA INFORMATION ENTRY AREA
IPLINFOA DS    CL(IPLLEN)              IPL INFORMATION ENTRY AREA
SYSINFOA DS    CL(IEASYSLN)            IEASYSNN INFORMATION MEMBER
```

```
*                                      INFORMATION AREA
WORKALEN EQU   *-WORKAREA              WORK AREA LENGTH
         TITLE 'PSA DSECT'
         IHAPSA DSECT=YES,LIST=NO
         TITLE 'CVT DSECT'
         CVT   PREFIX=YES,DSECT=YES,LIST=YES
         TITLE 'EXTENDED CVT'
         IHAECVT DSECT=YES,LIST=NO
         TITLE 'IPA'
         IHAIPA
         TITLE 'PCCAT'
         IHAPCCAT DSECT=YES
         TITLE 'PCCA'
         IHAPCCA DSECT=YES
         TITLE 'SERVICE CALL CONTROL BLOCK'
         IHASCCB
         TITLE 'TSVT'
         IKJTSVT
         TITLE 'SMF SMCA'
         IEESMCA
         TITLE 'UCB'
UCB      DSECT
         IEFUCBOB
         TITLE 'MVS INFORMATION'
MVSI     DSECT
MPRODO   DS    CL16              PRODUCT OWNER
MPRODNM  DS    CL16              PRODUCT NAME
MPRODVER DS    CL2               PRODUCT VERSION
MPRODREL DS    CL2               PRODUCT RELEASE
MPRODMOD DS    CL2               PRODUCT MODIFICATION
MPRODN   DS    CL8               PRODUCT NAME OF THE CONTROL PROGRAM
MPRODI   DS    CL8               PRODUCT FMID
MIPLTIME DS    XL4               IPL TIME
MIPLDATE DS    CL4               IPL DATE
IPLVOL   DS    CL6               IPL VOLUME SERIAL NO
MVSILEN  EQU   *-MVSI            MVS INFORMATION LENGTH
         TITLE 'CPU INFORMATION'
CPUINF   DSECT
PCPID    DS    XL12              CPU ID + SERAL NUMBER(PCCACPID)
PCPUA    DS    XL2               CPU ADDRESS(PCCACPUA)
PPSAV    DS    XL4               VIRTUAL ADDRESS OF PSA(PCCAPSAV)
PPSAR    DS    XL4               REAL ADDRESS OF PSA(PCCAPSAR)
PISCM    DS    X                 INTERRUPT SUB-CLASS MASK(PCCAISCM)
SCPA     DS    X                 CPU ADDRESS
STOD#    DS    X                 TOD NUMBER
SCPFL    DS    X                 CPU CHARACTERISTICS FLAG 1
SCPF2    DS    X                 CPU CHARACTERISTICS FLAG 2
CPULEN   EQU   *-CPUINF          CPU ENTRY LENGTH
CPUMLEN  EQU   (CPULEN*16)       MAXIMUM NUMBER OF ENTRIES
         TITLE 'IPL INFORMATION'
IPLINF   DSECT
LPARM    DS    CL8               LOAD PARM
IODFU    EQU   LPARM,4           IODF UNIT ADDRESS
```

```
LOADS    EQU   LPARM+4,2         LOADXX SUFFIX
PROMT    EQU   LPARM+6,1         OPERATOR PROMPT
NUCID    EQU   LPARM+7,1         NUCLEUS ID
IODFHLQ  DS    CL8               IODF HLQ
IODFSUF  DS    CL2               IODF SUFFIX
IOCFG    DS    CL8               IO CONFIGURATION ID
IOEDT    DS    CL2               IO EDT
NUCLSTID DS    CL2               NUCLST ID
SYSPARM  DS    CL63              LIST OF IEASYS SUFFIXES
IEASYM   DS    CL63              LIST OF IEASYM SUFFIXES
SYSPLEX  DS    CL8               SYSPLEX NAME
LPARNAME DS    CL8               LPAR NAME
HWNAME   DS    CL8               HARDWARE NAME
MCATD    DS    CL44              MASTER CATALOG DSNAME
MCATV    DS    CL6               MASTER CATALOG VOLUME
MCATT    DS    CL1               MASTER CATALOG TYPE
MCATAL   DS    CL1               ALIAS NAME LEVEL
CCAS     DS    CL2               CAS SERVICE TASK LOWER LIMIT
PLIBDSN  DS    CL44              IPL PARMLIB DSN
PLIBDDV  DS    CL4               IPL PARMLIB DEVICE NO
IODDS    DS    CL1               DEVICE SUPPORT MODULES
PARMDSN  DS    CL44              PARMLIB DSN
PARMVOL  DS    CL6               PARMLIB VOLSER
PARMFLAG DS    X                 PARMLIB USAGE FLAG
IPLLEN   EQU   *-IPLINF          IPL ENTRY LENGTH
         TITLE 'HSA INFORMATION'
HSAI     DSECT
HSAZ     DS    XL2               HSA SIZE
HSAA     DS    XL4               HSA ADDRESS
HSALEN   EQU   *-HSAI            HSA ENTRY LENGTH
HSAMLEN  EQU   (HSALEN*16)       MAXIMUM NUMBER OF ENTRIES
         TITLE 'COMMON SCP INFORMATION'
CSCP     DSECT
*
*        COMMAND DEPENDENT DATA FROM
*        READ SCP INFO COMMAND.
*
SCPI     DS    ØF                MAPPING OF SCCB COMMAND DEPENDENT DATA
*                                FIELD, SCCBCMDD, FOR SERVICE PROCESSOR
*                                COMMAND READ SCP INFO.
SAR      DS    XL2               REAL STORAGE ADDRESS RANGE. MAXIMUM
*                                STORAGE INCREMENT NUMBER INSTALLED.
SAI      DS    XL1               REAL STORAGE ADDRESS INCREMENT,
*                                IN UNITS OF 1M.
SBS      DS    XL1               REAL STORAGE BLOCK SIZE IN UNITS OF 1K
SII      DS    XL2               REAL STORAGE INCREMENT
*                                BLOCK INTERLEAVE INTERVAL.
NCPS     DS    H                 NUMBER OF CPUS INSTALLED.
NHSA     DS    H                 NUMBER OF HSAS.
PARM     DS    CL8               LOAD PARAMETER INFORMATION FROM
*                                SERVICE PROCESSOR.
MESI     DS    XL4               EXTENDED STORAGE ADDRESS RANGE.
*                                MAXIMUM EXTENDED STORAGE INCREMENT
*                                NUMBER INSTALLED.
NXSB     DS    XL4               NUMBER OF 4K STORAGE BLOCKS IN AN
*                                EXTENDED STORAGE INCREMENT.
MESE     DS    XL2               MAXIMUM EXTENDED STORAGE ELEMENT
*                                NUMBER INSTALLED.
VPRM     DS    ØF                VECTOR PARAMETERS.
VSS      DS    XL2               VECTOR SECTION SIZE.
VPSM     DS    XL2               VECTOR PARTIAL SUM NUMBER.
IFM      DS    ØCL8              INSTALLED FACILITY MAP.
IFM1     DS    CL1               INSTALLED FACILITY MAP BYTE 1.
CHPI     EQU   X'8Ø'             CHANNEL PATH INFORMATION INSTALLED.
CHPS     EQU   X'4Ø'             CHANNEL PATH SUBSYSTEM COMMAND
*                                INSTALLED.
CHPR     EQU   X'2Ø'             CHANNEL PATH RECONFIGURATION
*                                INSTALLED.
CPUI     EQU   X'Ø8'             CPU INFORMATION INSTALLED.
CPUR     EQU   X'Ø4'             CPU RECONFIGURATION INSTALLED.
IFM2     DS    CL1               INSTALLED FACILITY MAP BYTE 2.
SGNL     EQU   X'8Ø'             SIGNAL ALARM INSTALLED.
OMR      EQU   X'4Ø'             WRITE OPERATOR MESSAGE AND READ
*                                OPERATOR RESPONSE INSTALLED.
STST     EQU   X'2Ø'             STORE STATUS ON LOAD INSTALLED.
RSTR     EQU   X'1Ø'             RESTART REASONS INSTALLED.
ITRC     EQU   X'Ø8'             INSTRUCTION ADDRESS TRACE BUFFER
*                                INSTALLED.
LPRM     EQU   X'Ø4'             LOAD PARAMETER INSTALLED.
WDAT     EQU   X'Ø2'             READ AND WRITE DATA INSTALLED.
IFM3     DS    CL1               INSTALLED FACILITY MAP BYTE 3.
SIR      EQU   X'8Ø'             REAL STORAGE INCREMENT
*                                RECONFIGURATION INSTALLED.
SEI      EQU   X'4Ø'             REAL STORAGE ELEMENT INFORMATION
*                                INSTALLED.
SER      EQU   X'2Ø'             REAL STORAGE ELEMENT RECONFIGURATION
*                                INSTALLED.
CARS     EQU   X'1Ø'             COPY AND REASSIGN STORAGE INSTALLED.
SUM      EQU   X'Ø8'             EXTENDED STORAGE USABILITY MAP
*                                INSTALLED.
ESEI     EQU   X'Ø4'             EXTENDED STORAGE ELEMENT INFORMATION
*                                INSTALLED.
ESER     EQU   X'Ø2'             EXTENDED STORAGE ELEMENT
*                                RECONFIGURATION INSTALLED.
CARL     EQU   X'Ø1'             COPY AND REASSIGN STORAGE LIST
*                                INSTALLED
IFM4     DS    CL1               INSTALLED FACILITY MAP BYTE 4
VFR      EQU   X'8Ø'             VECTOR FEATURE RECONFIGURATION
*                                INSTALLED.
EVNT     EQU   X'4Ø'             READ / WRITE EVENT FEATURE
*                                INSTALLED.
RRGI     EQU   X'Ø8'             READ RESOURCE GROUP INFORMATION
*                                INSTALLED.
CON1     DS    CL1               BITS Ø-7 OF CONFIGURATION
*                                CHARACTERISTICS.
BBFY     EQU   X'8Ø'             CONFIGURATION IS RUNNING UNDER BFY.
```

```
SOPF     EQU   X'2Ø'              SUPPRESSION ON PROTECTION FACILITY
IRIN     EQU   X'1Ø'              INITIATE RESET INSTALLED
CSCF     EQU   X'Ø8'              STORE CHANNEL SUBSYSTEM
*                                 CHARACTERISTICS FACILITY IS INSTALLED
CON2     DS    CL1                BITS 8-15 OF CONFIGURATION
*                                 CHARACTERISTICS
CSLO     EQU   X'4Ø'              CSLO IS INSTALLED
ETR      DS    XL4                ETR-SYNC-CHECK TOLERANCE
CSCPLEN  EQU   *-CSCP             COMMON SCP ENTRY LENGTH
         TITLE 'CPU INFORMATION ENTRY'
BCP      DSECT                    CPU INFORMATION ENTRY.
BCPENTS  DS    XL2                NO OF BCP ENTRIES
*
*        ARRAY OF CPU INFORMATION FROM READ SCP INFO COMMAND.
*        (SCCBNCPS ENTRIES. ENTRIES BEGIN AT ADDR(SCCB)+SCCBOCP.)
*
CPA      DS    XL1                CPU ADDRESS.
TOD#     DS    XL1                TOD CLOCK NUMBER FOR THIS CPU.
CPFL     DS    XL1                CPU CHARACTERISTIC FLAGS BYTE 1.
*                                 (BIT POSITIONS 32-39.)
VFIN     EQU   X'8Ø'              VECTOR FEATURE INSTALLED.
VFCN     EQU   X'4Ø'              VECTOR FEATURE CONNECTED.
VFSB     EQU   X'2Ø'              VECTOR FEATURE IN STANDBY STATE.
CRIN     EQU   X'1Ø'              CRYPTO FEATURE INSTALLED.
CPF2     DS    XL1                CPU CHARACTERISTIC FLAGS BYTE 2.
*                                 (BIT POSITIONS 4Ø-47.)
MPSB     EQU   X'8Ø'              PRIVATE SPACE BIT IS INSTALLED.
PER2     EQU   X'Ø1'              PER 2 IS INSTALLED.
KSID     EQU   X'Ø1'              KSU ID OF INSTALLED CRYPTO FEATURE.
         TITLE 'ARRAY OF HSA INFORMATION'
HSA  DSECT                        HSA INFORMATION ENTRY.
*
*        ARRAY OF HSA INFORMATION FROM READ SCP INFO COMMAND.
*        (SCCBNHSA ENTRIES. ENTRIES BEGIN AT ADDR(SCCB)+SCCBOHSA.)
*
HASENTS  DS    XL2                NO OF HSA ENTRIES
HSSZ     DS    XL2                SIZE OF THIS HSA IN UNITS OF 4K.
AHSA     DS    XL4                ADDRESS OF THIS HSA.
         TITLE 'IEASYSXX INFORMATION'
IEASYSXX DSECT
PARMLEN  DS    XL2                LENGTH
SOURCE   DS    CL2                SOURCE OF PARAMETER STRING
PARMSTRG DS    CL256              PARAMETER STRING INFORMATION
IEASYSLN EQU   *-PARMLEN          MEMBER INFORMATION MAX LEN
         END   SCPINFO
```

Editor's note: this article will be continued in the next issue.

Rem Perretta
Senior Systems Programmer
Millenium Computer Consultancy (UK)

Can the Internet handle mainframe volumes?

INTRODUCTION

While testing a new version of a product from a third-party software supplier, a problem arose that resulted in the generation of a dump. After discussing the problem with the technical support hotline, they requested that I send them the dump for further analysis.

The normal procedure would be to copy the data onto a cartridge and send this to their offices, using an overnight courier service. As I knew that this supplier had a World Wide Web presence, as seems to be the case with most companies these days, I decided to investigate the possibility of using an Internet File Transfer to get the dump to them.

It turned out that this was indeed a possibility: they had a File Transfer Protocol (FTP) server at their site, so all I had to do was to connect to their Internet site with an FTP client and issue a PUT command, then sit back and watch the bits fly.

PROBLEMS ENCOUNTERED

It all sounded so easy in theory, but in practice I discovered that several crucial links in the chain were not designed for the kind of application I was attempting.

The major problems lay with the size of the file that I was trying to transfer. It was a SYSUDUMP, which had been written to the JES SYSOUT class, and then copied to a VBA disk dataset using the SDSF PRINT ODSN command. The dump was 1,175,174 lines in length, and when copied to disk with DCB parameters RECFM=VBA, LRECL=133 and BLKSIZE=27998, it consumed 2,670 tracks on a 3390. This is approximately 150MB, and while this is not huge, it still presented many logistical problems.

File compression

My first goal was to reduce this as much as possible while still having the power of the mainframe at my disposal, and the level of compression that I achieved using the TRSMAIN utility seemed extremely impressive, until it occured to me that a SYSUDUMP often contains long strings of repeating zeros (X'F0'), as well as a great deal of space characters (X'40') because it is being formatted for printing.

Nevertheless, I reduced the 150MB file to a RECFM=FB, LRECL=1024, BLKSIZE=6144 file of 780 tracks on a 3390, or around 38MB. The job consumed 234 seconds of CPU on an IBM CMOS-based system, and the elapsed time was approximately 10 minutes with a mix of other on-line and batch tasks running, so the compression process seems very CPU intensive.

The site where I am working has no host-based Internet FTP capability, so I intended to use a PC with a 56KB dial-up connection to an Internet Service Provider as my FTP platform. A quick calculation showed me that the 38MB over a 56KB modem would take a minimum of around 90 minutes (5558 seconds at 7KB per second) to transfer. This seemed acceptable because it was certainly faster than an overnight courier.

File transfer to the PC

But of course I am jumping the gun, as I still did not have the file on the PC platform. The only utility available was the IBM 3270 PC File Transfer Program for MVS/TSO Release 1.1.1, better known as IND$FILE, which as anyone who has tried it for large files will know, is not very fast. My PC was running IBM Personal Communications 3270 Version 4.1 for Windows with an IEEE 802.2 connection to the host.

This download ran at a rate of around 6KB per second, so the 38MB file took around 2 hours in total, but now I had the data safely on the office file server and accessible by any other PC attached to the LAN.

And therein lay a major problem – it was company policy for any PC which had the capability of connecting to the Internet not to be connected to the LAN. This was neccessary to prevent the infection of the entire company network with viruses and the like.

This immediately brought home to me an aspect of the PC industry that lags far behind the amazing advances we have seen in processor and disk technology, namely removable media or floppy disk drives.

There are several competing technologies seeking to be the floppy of the 21st century, which is probably part of the problem in that no single standard has emerged to replace the ubiquitous 1.44MB disk. An Iomega ZIP drive, an LS-120, even a recordable CD would have been lifesavers, but I was stuck with 1.44MB disks.

Not to be daunted, I figured that I could copy the file over multiple disks by using PKZIP. Of course the file was already compressed so PKZIP would not help there, and so I would need around 27 disks in total. This was the next problem. I did not have 27 disks, at least not without scrounging around and digging out dubious disks which have lain at the bottom of desk drawers for years. But then it occurred to me that as I was merely using the ZIP file as a transport mechanism, I only really needed one disk. I could run the ZIP function on my PC and the unZIP function on the target machine simultaneously, feeding the disk back and forth. The entire ZIPfile need never exist.

This method quickly ran into another annoying problem in that the unZIP function of PKZIP requires that you load the last disk of a multiple disk ZIP file immediately after beginning, presumably since critical information regarding how to unZIP the file can only be tagged on after the file has been fully zipped. The only way I could figure to get around this was to run the ZIP function twice; once just to create the final disk, then again for the simultaneous ZIP/unZIP. This worked perfectly, albeit requiring two disks instead of one, and a certain amount of walking between the two PCs, which were fortunately relatively close together.

Using FTP

At last I was ready to FTP the file to the supplier, and, although several hours had passed since I began, the end seemed to be in sight. I dialled the ISP and loaded Microsoft Internet Explorer, which I knew could handle FTP since I had used it often to download from all manner of Internet FTP sites. Of course I had never used it to send a file, but it did not occur to me that it was a one-way FTP utility. Another hour or so passed while I tried to figure out how to do this, finally contacting Microsoft technical support to discover that it could not be done. But no matter, I had another command line based FTP utility program available, and I was thrilled when this connected to my supplier’s FTP server at the very first attempt.

I issued a PUT command, and at last the file was actually transmitting. The transfer speed seemed somewhat slower than the 56KB modem should have been able to deliver, and it turns out that this is a manifestation of an interesting property of these devices, namely that a 56KB transfer rate is nigh impossible to achieve. This connection was running at closer to 28KB, so I recalculated the probable time for the transfer as 180 minutes.

Everything seemed to be OK until the transfer reached approximately 2MB, at which time the FTP session suddenly terminated with an unhelpful error message. I restarted the session, hoping that it would at least resume from where it left off, but it went back to the beginning, and again after 2MB it terminated. I suspected that the ISP account that I was using had a limit on uploading, and a call to their Help Desk confirmed this to be the case.

Since this was a problem beyond my technical ability to solve, I finally had to concede, and beat a tactical retreat by creating a cartridge just as I have done for the last 10 years. But I still had the final moral victory, as I found a colleague who had access to an alternative ISP and we successfully transferred the file, and the dump was available for analysis by the supplier before the cartridge arrived after all.

CONCUSIONS

Certainly, there are other, far more powerful, proprietary FTP solutions available, but in the final analysis I feel that this was a very useful exercise in using lowest common denominator-type methods. Only a couple of years ago it would have been impossible to even attempt such a task, today it can be done at a stretch, but I think that for the next such dump I have to send I will return to the legacy technique. But with new technologies increasing the speed of communications to the extent that LAN-speed Internet access may soon be commonplace, and a standard replacement for the 1.44MB disk surely imminent, sending cartridges may soon be history.

Patrick Mullen
MVS Systems Consultant (Canada)

October 1995 – September 1998 index

Items below are references to articles that have appeared in *MVS Update* since October 1996. References show the issue number followed by the page number(s). Back issues of *MVS Update* are available back to Issue 100 (January 1995).

MVS news

IBM has launched Version 3 Release 1 of its ADSM for MVS, with central management of multiple ADSM servers, lights-out server automation, and remote Help Desk support. There is a new ADSM enterprise console, a Web-based interface, which extends the capabilities of the current storage administrator interface. Also, with the optional Enterprise Administration feature, the console provides a more global scope. A new Web back-up-archive client allows an authorized user to perform back-up and restore functions while operating remotely.

The enterprise console will act as the integration point for all ADSM functions and commands, and it includes the Web administrative client interface and a Web back-up-archive client interface. It lets administrators navigate, logon, and perform functions on any ADSM Version 3 Release1 server or Web client from a supported Web browser.

Activities include server administration, client operations through the Web back-up-archive client interface, unified login for all functions, monitoring of all client/server events to forward all events to the primary configuration/event server, and all regular administration functions available in Version 3 Release 1.

The new Web back-up-archive client interface, meanwhile, will let the administrator or Help Desk connect to any remote client and perform GUI operations such as back-up-archive and restore-retrieve, on behalf of the end-user.

* * *

IBM has announced Release 14 of its DFSORT sort, merge, copy, analysis and reporting option for OS/390 and MVS/ESA, including a range of enhancements for productivity, performance, capacity, and storage usage.

Among these are symbols for fields and constants in DFSORT and ICETOOL statements. Users can create and use symbols for their own data, and use symbols from IBM for data associated with RACF, DFSMSrmm, and DCOLLECT.

A time-of-day installation option control lets you adjust the resources available for DFSORT applications according to the day and time they run, and the product allows sorting and merging of larger datasets.

There is also simplified installation and customization, with fewer FMIDs and libraries, and the ability to replace IEBGENER with ICEGENER more easily. And it gets improved performance, and storage usage and virtual storage constraint relief for copy, merge, and ICEGENER applications.

More INCLUDE/OMIT conditions and SUM fields allow users to write more complex filtering and totalling applications, while new OUTFIL features support multiple output records using the fields of each input record, split records, double and triple space in reports, and pad short fields.

Contact your local IBM representative for further information.

* * *