137

VM

update

January 1998

In this issue

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38030
From USA: 01144 1635 38030
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

Australian office
Xephon/RSM
GPO Box 6258
Halifax Street
Adelaide, SA 5000
Australia
Telephone: 088 223 1391

Editorial panel
Articles published in *VM Update* are reviewed by our panel of experts. Members of the panel include John Illingworth (UK), Reinhard Meyer (Germany), Philippe Taymans (Belgium), Romney White (USA), Martin Wicks (UK), and Jim Vincent (USA).

Contributions
Articles published in *VM Update* are paid for at the rate of £170 ($250) per 1000 words for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

Editor
Robert Burgess

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.

Subscriptions and back-issues
A year's subscription to *VM Update*, comprising twelve monthly issues, costs £170.00 in the UK; $255.00 in the USA and Canada; £176.00 in Europe; £182.00 in Australasia and Japan; and £180.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1990 issue, are available separately to subscribers for £14.50 ($21.50) each including postage.

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

Printed in England.

Enhanced sort routine

GENERAL DESCRIPTION

ADSORT enhances the XEDIT SORT macro by allowing users to specify the sort order for each sorted field in a sorted file record. The sort is done in EBCDIC order.

ADSORT is written in Assembler and REXX and runs under CMS with VM/SP Release 5.

MEMORY REQUIREMENTS

To accelerate read/write operations, a 128KB buffer area is allocated in sort exit module ADSORTA. The size of ADSORTA is 784 bytes.

ADSORT USAGE

The ADSORT macro is invoked as shown below:

```
ADSORT <target> <A|D> <FB1 FE1> [<A|D> <FB2 FE2>]
```

where:

- 'Target' specifies the number of records to be sorted. It is processed by the LOCATE subcommand, and may specify a relative displacement from the current line, a line number, or a string target. The string target must be delimited by '/' and may be preceded by '+' or '-'.
- 'A' sorts the fields following the command, in ascending order.
- 'D' sorts the fields following the command, in descending order.
- 'FBn FEn' determines the nth sorted field, where FBn and FEn are the starting and the ending columns of a sort field, and n is between 1 and 10 inclusive.

If the same sort order (ascending or descending) is specified for all

sorted fields, then the CMS SORT macro runs directly from the ADSORT macro, and sort exit (ADSORTA) is not activated. Otherwise, ADSORTA is invoked before and after execution of CMS SORT.

Because the code of ADSORTA is reduced and optimized, the sort time may increase slightly when large files are processed.

Examples of ADSORT usage are:

- Sort the file from the current line to the top of the file:

```
ADSORT -* D 1 2 A 3 4
```

- Sort five lines from the current line towards the end of the file:

```
ADSORT 5 A 1 2 D 3 4
```

- Sort seven lines from the current line backwards to the top of the file:

```
ADSORT -7 D 1 2 A 3 4 D 5 6
```

- Sort the file from the current line to line nine:

```
ADSORT :9 A 1 2 D 3 4 A 5 6 D 7 8
```

INSTALL EXEC

```
/*********************************************************************/
/***                                                    ***         ***/
/*** INSTALL          generate ADSORTA MODULE          ***  DG'97  ***/
/***                                                    ***         ***/
/*********************************************************************/
/***   SIZE ØØØ31  VER 1.Ø MOD ØØ  TIME 16:48:Ø1  DATE Ø8/Ø8/97   ***/
/*********************************************************************/

  CLRSCRN
  MESSAGE = 'user request'
  SAY ' --- Start ADSORTA MODULE generation - reply Y or N'
  PULL REPLY
  IF REPLY ¬= 'Y' THEN
  SIGNAL ERROR
  SET CMSTYPE HT
  SIGNAL ON ERROR
  MESSAGE = 'error when assemble' ADSORTA
  ASSEMBLE ADSORTA
  ERASE ADSORTA LISTING A
  MESSAGE = 'error when load' ADSORTA
  LOAD ADSORTA '(' NOMAP NOLIBE
```

```
  MESSAGE = 'error when genmod' ADSORTA
  GENMOD
  ERASE ADSORTA TEXT A
  SIGNAL OFF ERROR
  SET CMSTYPE RT
  SAY ' --- ADSORTA MODULE generated successfully'
  EXIT
ERROR:
  SET CMSTYPE RT
  SAY ' --- ADSORTA MODULE not generated because of' MESSAGE
```

ADSORTA ASSEMBLE

```
*********************************************************************
****                                                  ***         ****
**** ADSORTA             XEDIT mixed sort            ***  DG'97  ****
****                                                  ***         ****
*********************************************************************
****   SIZE ØØ127  VER 1.Ø MOD ØØ  TIME 17:ØØ:54  DATE Ø8/Ø8/97   ****
*********************************************************************
*                                                                   *
ADSORTA  CSECT
         USING *,12
         LR    11,14
         FSREAD 'ADSORT PARAM A',BSIZE=6Ø,BUFFER=GETPARAM,RECNO=1
         FSCLOSE 'ADSORT PARAM A'
         LA    1Ø,GETPARAM
         USING PARAM,1Ø
         MVC   FSCBFN(18),FNAME
         MVC   FSCBANIT(4),TOSORT
         LH    9,LRECL
         SR    Ø,Ø
         L     1,=F'131Ø65'
         DR    Ø,9
         C     1,FSCBANIT
         BH    ALLOC
         ST    1,FSCBANIT
ALLOC    EQU   *
         L     1,FSCBANIT
         MH    1,LRECL
         ST    1,FSCBSIZE
         LA    Ø,7(1)
         SRL   Ø,3
         LR    8,Ø
         DMSFREE DWORDS=(Ø),TYPE=USER,ERR=RET
         ST    1,FSCBBUFF
         SR    1,1
         IC    1,TOGEN
```

```
         LA    2,DATA
         LA    3,GENAREA
DOGEN    EQU   *
         MVI   Ø(3),X'D7'
         MVC   1(1,3),1(2)
         MVI   2(3),X'1Ø'
         MVC   3(1,3),Ø(2)
         MVC   4(2,3),=X'2ØØØ'
         LA    2,2(2)
         LA    3,6(3)
         BCT   1,DOGEN
         MVC   Ø(MOVELEN,3),GENEND
         LA    2,FILLER
         L     7,TOSTART
         SR    6,6
CYC      EQU   *
         ST    7,FSCBAITN
         MVC   FSCBCOMM(8),=CL8'DMSXFLRD'
         LA    Ø,EXTPLIST
         LA    1,FSCB
         ICM   1,8,=X'Ø2'
         SVC   2Ø2
         DC    AL4(1)
         LTR   15,15
         BNZ   FREEMEM
         L     Ø,FSCBANIT
         L     1,FSCBBUFF
         AR    7,Ø
         AR    6,Ø
GENAREA  EQU   *
         DC    34X'Ø7ØØ'
JUMP     EQU   *
         MVC   FSCBCOMM(8),=CL8'DMSXFLWR'
         LA    Ø,EXTPLIST
         LA    1,FSCB
         ICM   1,8,=X'Ø2'
         SVC   2Ø2
         DC    AL4(1)
         L     5,TOSORT
         SR    5,6
         BZ    FREEMEM
         C     5,FSCBANIT
         BH    CYC
         ST    5,FSCBANIT
         MH    5,LRECL
         ST    5,FSCBSIZE
         B     CYC
FREEMEM  EQU   *
         LR    Ø,8
         L     1,FSCBBUFF
```

```
         DMSFRET DWORDS=(Ø),LOC=(1)
         SR    15,15
RET      EQU   *
         BR    11
GENEND   AR    1,9
         BCT   Ø,GENAREA
         B     JUMP
MOVELEN  EQU   *-GENEND
         DS    ØF
GETPARAM EQU   *
         DS    CL6Ø
EXTPLIST EQU   *
         DC    A(COMMVERB)
         DC    A(Ø)
         DC    A(Ø)
         DC    A(Ø)
COMMVERB DC    CL8'SUBCOM'
FSCB     EQU   *
FSCBCOMM DC    CL8' '
FSCBFN   DC    CL8' '
FSCBFT   DC    CL8' '
FSCBFM   DC    CL2' '
FSCBITNO DC    H'Ø'
FSCBBUFF DC    A(Ø)
FSCBSIZE DC    A(Ø)
FSCBFV   DC    CL1'F'
FSCBFLG  DC    X'ØØ'
FSCBNOIT DC    H'ØØ'
FSCBNORD DC    AL4(Ø)
FSCBAITN DC    AL4(Ø)
FSCBANIT DC    AL4(Ø)
FSCBWPTR DC    A(Ø)
FSCBRPTR DC    A(Ø)
FILLER   DC    128X'FF'
PARAM    DSECT
FNAME    DS    18C
LRECL    DS    2C
TOSTART  DS    4C
TOSORT   DS    4C
TOGEN    DS    C
DATA     DS    C
         END   ADSORTA
```

ADSORT XEDIT

```
/*********************************************************************/
/***                                                    ***         ***/
/*** ADSORT              XEDIT mixed sort                 ***  DG'97  ***/
/***                                                    ***         ***/
```

```
/*********************************************************************/
/***   SIZE 00088  VER 1.0 MOD 01  TIME 16:52:48  DATE 08/08/97   ***/
/*********************************************************************/

  PARSE ARG PARAMETER_STRING

  I = INDEX(PARAMETER_STRING, '/') + 1
  IF I > 1 THEN
  DO
    TARGET = SUBSTR(PARAMETER_STRING, 1, POS('/', PARAMETER_STRING,I))
    PARAM = SUBSTR(PARAMETER_STRING, POS('/', PARAMETER_STRING, I)+1)
  END
  ELSE
  PARSE VAR PARAMETER_STRING TARGET PARAM
  TO_GEN = 0
  PARSE VAR PARAM                                                    ,
      O.1 B.1 E.1 O.2 B.2 E.2 O.3 B.3 E.3 O.4 B.4 E.4 O.5 B.5 E.5    ,
      O.6 B.6 E.6 O.7 B.7 E.7 O.8 B.8 E.8 O.9 B.9 E.9 O.10 B.10 E.10 .
  DO I = 1 TO 10
    IF O.I = '' THEN
    LEAVE
    IF ¬ (O.I = 'A' | O.I = 'D') THEN
    DO
      EMSG '--- Invalid sort order defined'
      EXIT
    END
    IF  ¬ (DATATYPE(B.I) = 'NUM' & DATATYPE(E.I) = 'NUM') |          ,
        B.I > E.I | B.I = 0 | E.I > LRECL.1 THEN
    DO J = 1 TO I - 1
      IF B.I <= E.J THEN
        IF B.I >= B.J THEN
      DO
ERROR:
        EMSG '--- Invalid sort field defined'
        EXIT
      END
    END
    IF O.I = 'D' THEN
    DO
      TO_GEN = TO_GEN + 1
      BGN.TO_GEN = B.I
      END.TO_GEN = E.I
    END
  END
  IF TO_GEN = I - 1 THEN
  DO
    TO_GEN = 0
    TARGET = TARGET D
  END
  IF TO_GEN = 0 THEN
```

```
DO
  MACRO SORT TARGET TRANSLATE(PARAM, '', 'AD')
  EXIT
END
EXT'/FN/FT/FM/LI/LR/SIZ/CASE/RECF'
STAY = LINE.1
LOCATE TARGET
IF RC > 1 THEN
EXIT
EXT'/LI'
START = MAX(MIN(STAY, LINE.1), 1)
TO_SORT = MIN(MAX(STAY, LINE.1), SIZE.1) - START + 1
IF TO_SORT = Ø THEN
EXIT
AD_SORT_PARAM = LEFT(FNAME.1, 8)LEFT(FTYPE.1,8)LEFT(FMODE.1, 2)    ,
                     ||  RIGHT(X2C(D2X(LRECL.1)), 2, 'Ø'X)         ,
                     ||  RIGHT(X2C(D2X(START)), 4, 'Ø'X)           ,
                     ||  RIGHT(X2C(D2X(TO_SORT)), 4, 'Ø'X)         ,
                     ||  X2C(D2X(TO_GEN))
DO I = 1 TO TO_GEN
  AD_SORT_PARAM = AD_SORT_PARAM || X2C(D2X(BGN.I - 1))
  AD_SORT_PARAM = AD_SORT_PARAM || X2C(D2X(END.I - BGN.I))
END
CASE MIXED RESPECT
RECF F
'EXECIO 1 DISKW ADSORT PARAM A 1 (FINI ST' AD_SORT_PARAM
ADDRESS CMS ADSORTA
':'STAY
MACRO SORT TARGET TRANSLATE(PARAM, '', 'AD')
ADDRESS CMS ADSORTA
':'STAY
CASE CASE.1 CASE.2
RECF RECFM.1
ADDRESS CMS 'ERASE ADSORT PARAM A'
```

ADSORT PREPARATION

INSTALL EXEC should be used to generate the executable sort exit ADSORTA. Macro ADSORT may then be invoked from XEDIT to sort any file being edited. The user must specify the sort order for each sorted field.

Dobrin Goranov
Information Services Co (Bulgaria)

SQL/DS source management utility

SSM is a utility that was written to help our DBAs maintain their SQL/DS DDL source files.

Our installation has two SQL/DS machines running in VM and uses VSE guest sharing as well. The DBAs tried to keep all their CREATE TABLE statements in one place, that is in CMS files on a mini-disk. Unfortunately, many VSE jobs also used the same DDL statements, and so different versions of DDL statements existed for the same objects – the first step down the road to chaos.

Because of this, I decided to write the SSM utility, in which all DDL statements (in the format required by DBSU) are stored on a central mini-disk (one mini-disk per SQL/DS machine, one file per table).

These DDL files contain the following statements:

- ACQUIRE DBSPACE
- CREATE TABLE
- CREATE INDEX
- CREATE VIEW
- Other statements such as DATAUNLOAD and DATALOAD.

The files use a simple syntax, with keywords delimiting the different statements (see the following example file). The filename is an abbreviation of the table name (but could be anything), the filetype must be SQL.

These DDL statements can be easily and selectively extracted, both interactively (by means of a selection panel), and automatically (with a special include function for VSE jobs).

SSM consists of the following REXX procedures:

- SSMEXP REXX

  The ‘heart’ of SSM is a pipe stage, which extracts the DDL statements selected, with arguments, and writes them to its output

stream. It is never called directly by the user, but only by other procedures.

- SSMUTIL EXEC

  This is the interactive utility, which displays a panel where the user can enter his selection criteria. It then calls SSMEXP, and writes the resulting DBSU control statements to a CMS file.

- SSMPANEL IOS3270

  This is the panel used by SSMUTIL. We have IBM's IOS3270 program installed, but SSMUTIL could easily be modified to use some other dialog manager, such as ISPF.

- SSMINC REXX

  This pipe stage is used by our VSE submit procedure. It expands the special <SQL-INC> statements contained in a JCL file by calling SSMEXP REXX.

  It should be possible to build the call to SSMINC into any submit procedure.

There are several points to consider if you plan to use these procedures:

- SSMEXP REXX 'knows' only two databases/environments (SQLTEST/T and SQLPROD/P). These are hard-coded and will have to be adjusted to your requirements.
- SSMEXP REXX automatically generates a CONNECT statement with a hard-coded user-id of DBOWNER. This is because all our SQL/DS objects are owned by the same user-id.

Beneath the listings of these procedures, this article contains the following material:

- DEMOTAB SQL – a sample DDL source file.
- SAMPLE PANEL – the hardcopy of the panel presented by SSMUTIL, filled with some selection criteria.
- SSMSAMP1 DBSU – the DBSU control file produced by

SSMUTIL, based upon the selection criteria seen on SAMPLE PANEL.

- SSMSAMP2 JCL1 – a sample VSE job containing a <SQL-INC> statement.
- SSMSAMP2 JCL2 – the same job, after being processed by SSMINC.
- SUBMIT STUB – a few lines from our VSE submit procedure, containing the call to SSMINC.

SSMEXP REXX

```
/*====================================================================*/
/* Name        :  SSMEXP   REXX                                       */
/*====================================================================*/
/* Application :  SSM                                                 */
/* Created     :  19 Mar 1997                                         */
/*                                                                    */
/* Usage       :  Pipeline Stage Command                              */
/*                                                                    */
/* Arguments   :  environment filename commands                       */
/*                                                                    */
/* environment  -> T for SQLTEST, P for SQLPROD                       */
/*                                                                    */
/* filename     -> name of the file containing DDL statements         */
/*                 (filetype must be SQL)                             */
/*                                                                    */
/* commands     -> a string of blank delimited words which can have   */
/*                 any of the following values:                       */
/*                                                                    */
/*                    COMM    generates COMMIT WORK                   */
/*                    STAT    generates UPDATE STATISTICS             */
/*                    ERRON   generates SET ERRORMODE ON              */
/*                    ERROFF  generates SET ERRORMODE OFF             */
/*                    ERRCONT generates SET ERRORMODE CONTINUE        */
/*                    S-D     generates DROP DBSPACE                  */
/*                    S-C     generates ACQUIRE DBSPACE               */
/*                    S-DC    generates both                          */
/*                    T-D     generates DROP TABLE                    */
/*                    T-C     generates CREATE TABLE                  */
/*                    T-DC    generates both                          */
/*                    I-D     generates DROP INDEX for all indexes    */
/*                    I-C     generates CREATE INDEX                  */
/*                    I-DC    generates both                          */
```

```
/*                    V-D     generates DROP VIEW for all views       */
/*                    V-C     generates CREATE VIEW                   */
/*                    V-DC    generates both                          */
/*                    UTILnn  generates whatever is defined in the    */
/*                            DDL file under this keyword             */
/*                            (nn can be anything)                    */
/*                                                                    */
/* Result      :  -                                                   */
/*                                                                    */
/* Function    :  SQL/DS Source Manager (Source Expander)             */
/*                                                                    */
/* InStream  Ø :  -                                                   */
/* OutStream Ø :  requested SQL DDL Statements for DBSU               */
/*                                                                    */
/*====================================================================*/
arg env filename cmds

if env = 'T' then db='TESTDB'
if env = 'P' then db='PRODDB'

out.Ø=Ø

call read_source

call load_part 'space'
call load_part 'table'
call load_part 'index'
call load_part 'view'

call extract_names

conn='CONNECT DBOWNER' ,
     'IDENTIFIED BY DBOWNERPW' ,
     'TO' db  || ';'

'callpipe var conn | stem out. append'

stat='SET UPDATE STATISTICS OFF;'
'callpipe var stat | stem out. append'

do j=1 to words(cmds)
   cmd=word(cmds,j)
   select
     when cmd = 'STAT' then call process_statistics
     when cmd = 'COMM' then call process_commit
     when cmd = 'ERRON' then call process_errormode_on
     when cmd = 'ERROFF' then call process_errormode_off
```

```
      when cmd = 'ERRCONT' then call process_errormode_cont
      when left(cmd,4) = 'UTIL' then  call process_util
      when left(cmd,2) = 'S-' then  call process_space
      when left(cmd,2) = 'T-' then  call process_table
      when left(cmd,2) = 'I-' then  call process_index
      when left(cmd,2) = 'V-' then  call process_view
      otherwise call errmsg '====> invalid Keyword:' cmd
    end
end

'callpipe stem out. | *:'

return

/*====================================================================*/
/* read source file into stem source.                                 */
/*====================================================================*/
read_source:
select
  when env = 'T'
   then
   do
     machine='sqlefv'
     disk='198'
   end
  when env = 'P'
   then
   do
     machine='sysh'
     disk='191'
   end
  otherwise
   do
     call errmsg 'invalid environment:' env
     exit 12
   end
end

Vaddr_mode = acc_disk(machine,disk,'RR')

if words(vaddr_mode) < 2
then
   do
     call errmsg 'link error:' vaddr_mode
     exit 12
   end

fm=word(vaddr_mode,2)
```

```
ifid=filename 'SQL' fm

if x95Ø119(ifid) = Ø
then
   do
     call errmsg 'the file' ifid 'does not exist'
     call diskrel fm
     exit 12
   end

'callpipe <' ifid ,
      '| specs 1-72       1' ,
      '| nlocate 1-1 /*/' ,
      '| stem source.'

address cms 'release' fm '(detach'

return

/*====================================================================*/
/* load_part                                                          */
/*====================================================================*/
load_part:

arg part

from_label='<'part'>'
to_label='<'part'-END>'
ifid='x1bak sql a'
'callpipe    stem source.' ,
      '| between /'from_label'/ /'to_label'/' ,
      '| drop first 1',
      '| drop last  1',
      '| stem ' part'.'
return

/*====================================================================*/
/* extract names of objects                                           */
/*====================================================================*/
extract_names:

spaces.Ø=Ø
do i=1 to space.Ø
   if word(translate(space.i),1) = 'ACQUIRE'
   then
      do
        n=pos(' NAMED ',translate(space.i))
```

```
        spacename=word(substr(space.i,n+6),1)
        'callpipe var spacename | stem spaces. append'
      end
end


tables.Ø=Ø
do i=1 to table.Ø
   if word(translate(table.i),1) = 'CREATE' &,
      word(translate(table.i),2) = 'TABLE'
   then
      do
        filename=word(table.i,3)
        'callpipe var filename | stem tables. append'
      end
end


views.Ø=Ø
do i=1 to view.Ø
   if word(translate(view.i),1) = 'CREATE' &,
      word(translate(view.i),2) = 'VIEW'
   then
      do
        viewname=word(view.i,3)
        'callpipe var viewname | stem views. append'
      end
end


indices.Ø=Ø
do i=1 to index.Ø
   n=pos(' INDEX ',translate(index.i))
   if n > Ø
   then
      do
        indname=word(substr(index.i,n+6),1)
        'callpipe var indname | stem indices. append'
      end
end

return

/*====================================================================*/
/*  process util                                                      */
/*                                                                    */
/*  cmd is  UTILnn                                                    */
```

```
/*====================================================================*/
process_util:

call load_part cmd /* loads the  stem <util>.  */

'callpipe stem' cmd'. | count lines | var records'

if records > 0
then
  'callpipe stem' cmd'. | stem out. append'
else
  do
    err='==> No information for Keyword' cmd
    'callpipe var err | stem out. append'
  end

return


/*====================================================================*/
/* process space                                                      */
/*====================================================================*/
process_space:

parse value cmd with . '-' actions

if pos('D',actions) > 0
then
   'callpipe stem spaces.' ,
        '| specs /DROP DBSPACE / 1 1-15 next /;/ next' ,
        '| stem out. append'

if pos('C',actions) > 0
then
   'callpipe stem space. | stem out. append'

return

/*====================================================================*/
/* process table                                                      */
/*====================================================================*/
process_table:

parse value cmd with . '-' actions

if pos('D',actions) > 0
then
```

```
   'callpipe stem tables.' ,
        '| specs /DROP TABLE / 1 1-15 next /;/ next' ,
        '| stem out. append'

if pos('C',actions) > 0
then
   'callpipe stem table. | stem out. append'

return

/*====================================================================*/
/* process index                                                      */
/*====================================================================*/
process_index:

parse value cmd with . '-' actions

if pos('D',actions) > 0
then
   'callpipe stem indices.' ,
        '| specs /DROP INDEX / 1 1-15 next /;/ next' ,
        '| stem out. append'

if pos('C',actions) > 0
then
   'callpipe stem index. | stem out. append'

return

/*====================================================================*/
/* process view                                                       */
/*====================================================================*/
process_view:

parse value cmd with . '-' actions

if pos('D',actions) > 0
then
   'callpipe stem views.' ,
        '| specs /DROP VIEW / 1 1-15 next /;/ next' ,
        '| stem out. append'

if pos('C',actions) > 0
then
   'callpipe stem view. | stem out. append'

return

/*====================================================================*/
```

```
/* process statistics                                                    */
/*======================================================================*/
process_statistics:

'callpipe stem spaces.' ,
        '| specs /UPDATE ALL STATISTICS FOR DBSPACE / 1' ,
           '1-15 next /;/ next' ,
        '| stem out. append'

return
/*======================================================================*/
/* process commit                                                       */
/*======================================================================*/
process_commit:

'callpipe literal COMMIT WORK;' ,
        '| stem out. append'
return
/*======================================================================*/
/* process errormode on                                                 */
/*======================================================================*/
process_errormode_on:

'callpipe literal SET ERRORMODE ON;' ,
        '| stem out. append'

return
/*======================================================================*/
/* process errormode off                                                */
/*======================================================================*/
process_errormode_off:

'callpipe literal SET ERRORMODE OFF;' ,
        '| stem out. append'

return
/*======================================================================*/
/* process errormode continue                                           */
/*======================================================================*/
process_errormode_cont:

'callpipe literal SET ERRORMODE CONTINUE;' ,
        '| stem out. append'

return
/*======================================================================*/
/* error message to outstream                                           */
/*======================================================================*/
```

```
errmsg:

parse arg  msgtext
'callpipe var msgtext | *:'
return

/*====================================================================*/
/* acc_disk                                                           */
/*====================================================================*/
acc_disk:  procedure

ARG MACHINE,VADDR,MODE

call 'GETFMADR'
PULL . FMOD linkaddr .

address cms 'EXECIO 1 CP (STRING LINK' MACHINE VADDR LINKADDR MODE
LINKRC=RC

IF LINKRC <= 2
THEN
   DO
     LINKRC=Ø
     DO QUEUED()
        PULL .
     END
     'callpipe cms access' linkaddr fmod
     ret=linkaddr fmod
   END
ELSE
   DO
     SAY '****** Error linking to 'MACHINE VADDR
     say 'LINK' MACHINE VADDR LINKADDR MODE
     DO QUEUED()
        PULL MESSAGE
        SAY MESSAGE
     END
     SAY '****** RETURN CODE FROM LINK: 'LINKRC
     ret=linkrc
   END

RETURN ret
```

SSMUTIL EXEC

```
/*====================================================================*/
/* Name        :  SSMUTIL EXEC                                        */
/*====================================================================*/
```

```
/* Application :  SSM                                                 */
/* Created     :  19 Mar 1997                                         */
/*                                                                    */
/* Usage       :  Procedure                                           */
/*                                                                    */
/* Arguments   :  [ environment fn ft fm ]                            */
/*                                                                    */
/* Result      :  -                                                   */
/*                                                                    */
/* Function    :  SSM DBSU File Generator                             */
/*                                                                    */
/* This is the interactive SSM File Generator.                        */
/* When called without arguments, a panel is displayed prompting      */
/* the user for selection criteria; the requested DBSU statements     */
/* are then written to a CMS file.                                    */
/* When called with arguments, the file specified with arguments      */
/* fn ft fm is processed with the SSM include expander and written    */
/* to the file fn SSM A. This is to test JCL without submitting it    */
/*====================================================================*/
arg env fn ft fm

/*====================================================================*/
/* if arguments are given, the input file is processed directly       */
/* (calls ssminc)                                                     */
/*====================================================================*/
if env ¬= ''
then
   do
     ofid=fn 'ssm a'
     'pipe <' fn ft fm ,
        '| trunc 70' ,
        '| ssminc' env ,
        '| >' ofid
     'xedit' ofid
     exit
   end


/*====================================================================*/
/* inits                                                              */
/*====================================================================*/

cursor=''             /* for ios3270 */
message=''            /* for ios3270 */

write_mode='>'
cmd1=''
```

```
cmd2=''
cmd3=''
OFID='SSM       DBSU A'

/*====================================================================*/
/* main                                                               */
/*====================================================================*/

call panel 'ssmpanel'

if iosk =  'ENTER'
then
   do
     call generate
     'xedit' ofid
   end

return

/*====================================================================*/
/* generate                                                           */
/*====================================================================*/
generate:

'pipe ssmexp' env tabname cmd1 cmd2 cmd3 ,
    '| pad 8Ø |' write_mode ofid' fixed 8Ø'

return

/*====================================================================*/
/* panel processing                                                   */
/*====================================================================*/
panel:
arg panelid
if panelmsg = 'PANELMSG' then panelmsg=''
if cursor   = 'CURSOR'   then cursor=''
if cursor ¬= ''
then    do; options='(UPDATE 'CURSOR ; alarm='.A'; end
else    do; options=''               ; alarm=''  ; end
'IOS327Ø' panelid options
if rc ¬= Ø then do; say 'IOS327Ø Error:' rc; exit; end
panelmsg='';cursor='';return
```

SSMPANEL IOS3279

```
;*====================================================================*/
;* Name        :  SSMPANEL IOS327Ø                                    */
;*====================================================================*/
```

```
;* Application :  SSM                                                    */
;* Created     :  2 Nov 1996                                 */
;*                                                                       */
;* Usage       :  IOS327Ø Panel                                          */
;*                                                                       */
;* Arguments   :  -                                                      */
;*                                                                       */
;* Result      :  -                                                      */
;*                                                                       */
;* Function    :  Panel for SSMUTIL                                      */
;*                                                                       */
;*=======================================================================*/
.n
.y
.F F1 F2 F3 F4 F5 F6 F7 F8 F9 F1Ø F11 F12
.JX SET CTL . off
.jx set  normal intensity color=white
.jx set  high   intensity color=green
.JX SET CTL [ HIG=underline col=yel TYPE=(SKIP UNPROTECTED NULLS)
.JX SET CTL % col=red
.JX SET CTL # col=green
.JX SET CTL } col=green hig=rev
.e]
.&alarm
.c
}SSM           Generate DBSU Control File
]
 %&MELDUNG
]

Environment    [1&env    T or   P

Tablefile      [8&tabname

Commands       [5Ø&cmd1
               [5Ø&cmd2
               [5Ø&cmd3

  COMM     Commit Work                  UTILnn   Miscellaneous
  STAT     Update Statistics
  S-DC     DBSpace Drop/Create
  T-DC     Table   Drop/Create
  I-DC     Index   Drop/Create
  V-DC     View    Drop/Create
  ERRON    Set Errormode On
  ERROFF   Set Errormode Off
  ERRCONT  Set Errormode Continue

     Output File    [18&ofid
```

```
.b
               ENTER:Generate file                    PF3:Exit
```

SSMINC REXX

```
/*====================================================================*/
/* Name        :  SSMINC REXX                                         */
/*====================================================================*/
/* Application :  SSM                                                 */
/* Created     :  2Ø Mar 1997                                  */
/*                                                                    */
/* Usage       :  Pipeline Stage Command                              */
/*                                                                    */
/* Arguments   :  environment                                         */
/*                T for SQLTEST, P for SQLPROD                        */
/*                                                                    */
/* Result      :  -                                                   */
/*                                                                    */
/* Function    :  <SQL-INC> statement expander                        */
/*                                                                    */
/* InStream  Ø :  JCL with <SQL-INC> statements                       */
/* OutStream Ø :  JCL with expanded <SQL-INC> statements              */
/*                                                                    */
/* This is the SSM include expander. It passes everything from its    */
/* primary input stream to its primary output stream, with the        */
/* exception of records beginning with <SQL-INC> which are expanded   */
/* by calling the SSM expander.                                       */
/* The format of include records is as follows:                       */
/*       <SQL-INC> filename commands                                  */
/* The filename and commands are passed as arguments to the SSM      */
/* together with the environment parameter.                           */
/* There is no limit to the number of include records.                */
/*====================================================================*/
arg env
do forever
   'readto inrec'
   if rc <> Ø then leave
   if pos('<SQL-INC>',translate(inrec)) = Ø
   then
      'output' inrec
   else
      do
        arguments=substr(inrec,1Ø)
        'callpipe ssmexp ' env arguments ,
           '| *:'
      end
end

return
```

DEMOTAB SQL

```
* ====================================================================
*  DEMOTAB SQL
*
*  SSM SOURCE FILE FOR TABLE DEMOTAB
* ====================================================================

<SPACE>
ACQUIRE PUBLIC DBSPACE NAMED SDEMOTAB
                     (PAGES = 5ØØØ96,LOCK=ROW,STORPOOL=7) ;
<SPACE-END>

* ====================================================================

<TABLE>
CREATE TABLE DEMOTAB (
         RECID                CHAR(4)         NOT NULL,
         DEMSTAMM             INTEGER         NOT NULL,
         DEMVERKW             CHAR(24)        NOT NULL,
         DEMNHALB             INTEGER         NOT NULL,
         DEMVHALB             INTEGER         NOT NULL,
         DEMSATZA             CHAR(1)         NOT NULL,
         DEMSEQN1             CHAR(4)         NOT NULL,
         DEMSEQN2             INTEGER         NOT NULL,
         DEMNBUGJ             CHAR(1)         NOT NULL,
         DEMSOLHA             CHAR(1)         NOT NULL)
         IN SDEMOTAB;
COMMENT ON TABLE
   DEMOTAB IS 'DEMO TABLE DEMOTAB                          ';
GRANT ALL ON SQLEFV.DEMOTAB TO PUBLIC;
<TABLE-END>

* ====================================================================

<INDEX>
CREATE INDEX DEMOTAB_IX1 ON DEMOTAB (RECID  ASC);
CREATE INDEX DEMOTAB_IX2 ON DEMOTAB (DEMNHALB ASC);
CREATE INDEX DEMOTAB_IX3 ON DEMOTAB (DEMVHALB ASC);
<INDEX-END>

* ====================================================================

<UTILØ1>
SET AUTOCOMMIT ON;
```

```
DATALOAD TABLE (DEMOTAB) IF POS (257-257) = 'B'
         RECID                303-306   CHAR
         DEMSTAMM             1-4       FIXED
         DEMVERKW             5-28      CHAR
         DEMNHALB             29-32     FIXED
         DEMVHALB             33-36     FIXED
         DEMSATZA             37-37     CHAR
         DEMSEQN1             38-41     CHAR
         DEMSEQN2             42-45     FIXED
         DEMNBUGJ             258-258   CHAR
         DEMSOLHA             259-259   CHAR
INFILE (SQLLOAD BLKSZ(12844) PDEV(DASD) RECFM(FB) RECSZ(494))
        COMMITCOUNT(50000);
<UTIL01-END>
```

SAMPLE PANEL

```
SSM           Generate DBSU Control File


 Environment     p        T or   P

 Tablefile       demotab

 Commands        errcont s-dc t-c i-c stat



   COMM     Commit Work                   UTILnn   Miscellaneous
   STAT     Update Statistics
   S-DC     DBSpace Drop/Create
   T-DC     Table   Drop/Create
   I-DC     Index   Drop/Create
   V-DC     View    Drop/Create
   ERRON    Set Errormode On
   ERROFF   Set Errormode Off
   ERRCONT  Set Errormode Continue

      Output File     SSM       DBSU A

                ENTER:Generate file                    PF3:Exit
```

SSMSAMP1 DBSU

```
CONNECT DBOWNER IDENTIFIED BY DBOWNERPW TO PRODDB;
SET UPDATE STATISTICS OFF;
SET ERRORMODE CONTINUE;
DROP DBSPACE SDEMOTAB;
ACQUIRE PUBLIC DBSPACE NAMED SDEMOTAB
                     (PAGES = 500096,LOCK=ROW,STORPOOL=7) ;
```

```
CREATE TABLE DEMOTAB (
         RECID                CHAR(4)         NOT NULL,
         DEMSTAMM             INTEGER         NOT NULL,
         DEMVERKW             CHAR(24)        NOT NULL,
         DEMNHALB             INTEGER         NOT NULL,
         DEMVHALB             INTEGER         NOT NULL,
         DEMSATZA             CHAR(1)         NOT NULL,
         DEMSEQN1             CHAR(4)         NOT NULL,
         DEMSEQN2             INTEGER         NOT NULL,
         DEMNBUGJ             CHAR(1)         NOT NULL,
         DEMSOLHA             CHAR(1)         NOT NULL)
         IN SDEMOTAB;
COMMENT ON TABLE
   DEMOTAB IS 'DEMO TABLE DEMOTAB                          ';
GRANT ALL ON SQLEFV.DEMOTAB TO PUBLIC;
CREATE INDEX DEMOTAB_IX1 ON DEMOTAB (RECID  ASC);
CREATE INDEX DEMOTAB_IX2 ON DEMOTAB (DEMNHALB ASC);
CREATE INDEX DEMOTAB_IX3 ON DEMOTAB (DEMVHALB ASC);
UPDATE ALL STATISTICS FOR DBSPACE SDEMOTAB;
```

SSMSAMP2 JCL1

```
* $$ JOB JNM=SSMSAMP,CLASS=Ø,DISP=D
// JOB SSMSAMP
// EXEC PGM=ARIDBS,SIZE=AUTO
ØØØ13ØØØ
<SQL-INC> DEMOTAB T-C I-C STAT
/*
/&
* $$ EOJ
```

SSMSAMP2 JCL2

```
* $$ JOB JNM=SSMSAMP,CLASS=Ø,DISP=D
// JOB SSMSAMP
// EXEC PGM=ARIDBS,SIZE=AUTO
CONNECT DBOWNER IDENTIFIED BY DBOWNERPW TO PRODDB;
SET UPDATE STATISTICS OFF;
CREATE TABLE DEMOTAB (
         RECID                CHAR(4)         NOT NULL,
         DEMSTAMM             INTEGER         NOT NULL,
         DEMVERKW             CHAR(24)        NOT NULL,
         DEMNHALB             INTEGER         NOT NULL,
         DEMVHALB             INTEGER         NOT NULL,
         DEMSATZA             CHAR(1)         NOT NULL,
         DEMSEQN1             CHAR(4)         NOT NULL,
         DEMSEQN2             INTEGER         NOT NULL,
         DEMNBUGJ             CHAR(1)         NOT NULL,
         DEMSOLHA             CHAR(1)         NOT NULL)
```

```
         IN SDEMOTAB;
COMMENT ON TABLE
   DEMOTAB IS 'DEMO TABLE DEMOTAB                              ';
GRANT ALL ON SQLEFV.DEMOTAB TO PUBLIC;
CREATE INDEX DEMOTAB_IX1 ON DEMOTAB (RECID  ASC);
CREATE INDEX DEMOTAB_IX2 ON DEMOTAB (DEMNHALB ASC);
CREATE INDEX DEMOTAB_IX3 ON DEMOTAB (DEMVHALB ASC);
UPDATE ALL STATISTICS FOR DBSPACE SDEMOTAB;
/*
/&
* $$ EOJ
```

SUBMIT STUB

```
/*====================================================================*/
/* expand <SQL-INC> statements                                        */
/*                                                                    */
/* these statements could be inserted in a submit procedure           */
/*                                                                    */
/* variable JOBFILE contains id of JCL file to be submitted           */
/* variable ENV     contains T (for Test) or P (for Production)       */
/*====================================================================*/
'pipe <' jobfile '| locate 1-9 /<SQL-INC>/ | stem dummy.'
if dummy.0 > 0
then
     'pipe <' jobfile '| trunc 70' ,
            '| ssminc' env '| pad 80  | >' jobfile 'fixed 80'
```

Ch Hofstetter (Switzerland)

Subscribers who want copies of the code from this issue can call our Web site – www.xephon.com – and ask for the article they require. The article will then be e-mailed to them. This service is free to subscribers.

The first time you visit the site, you will need to register. For this, you will need the user-id, which appears on the label of the envelope in which you receive your issue of *VM Update*.

When you register, you will be asked to choose your own password and user name, which will be recognized when you log on again.

Dynamic menus system for CMS – part 4

This month we conclude the code for the on-line administration utilities that go with the dynamic menus system for CMS.

```
     /* BUILD PROCEDURE BODY */
      M=M+1
      LNKLINE.M=LEFT(N_PROC,9)||'LINK     '||LEFT(UID.N,9)||,
      RIGHT(O_ADDR.N,4,'Ø') RIGHT(T_ADDR.N,4,'Ø') LEFT(MODE.N,2) ,
      C2X(REVERSE(LPSWD.N))
      ACCLINE.M=LEFT(N_PROC,9)||'ACCESS   '||RIGHT(T_ADDR.N,4,'Ø') ,
      LEFT(A_MODE.N,3)
     END /* END N=1 TO UID.Ø */
     HEADLINE=LEFT(N_PROC,9)
     TDLINE=LEFT(N_PROC,9)||'TDISK    '||TDSK
     EXLINE=LEFT(N_PROC,9)||'EX       '||EXECNAME
     LNKLINE.Ø=M;ACCLINE.Ø=M
    'PIPE < MENU XSTATMS ',
    '| NLOCATE (1.8) /'STRIP(N_PROC)'/',
    '| APPEND VAR HEADLINE',
    '| APPEND STEM LNKLINE.',
    '| APPEND STEM ACCLINE.',
    '| APPEND VAR TDLINE',
    '| APPEND VAR EXLINE',
    '| > MENU XSTATMS A'
     PROC=N_PROC
    'XMITMSG 86 N_PROC (APPLID TAF CALLER XST NOCOMP VAR'
     UID.='        ';O_ADDR.='    ';T_ADDR.='    '
     MODE.='  ';LPSWD.='        ';A_MODE.='    ';UPDYN='N'
     SIGNAL LOAD_ARRAYS
    END /* END UPDYN=Y */
    SIGNAL XSTATMS
   END /* END ENTER */
WHEN VALUE(AIDKEY)='PF1Ø' THEN DO                  /* SCROLL BACKWARD */
     P=P-1; IF P<=Ø THEN P=PROCS.Ø
     UID.='        ';O_ADDR.='    ';T_ADDR.='    '
     MODE.='  ';LPSWD.='        ';A_MODE.='    ';UPDYN='N'
     SIGNAL LOAD_PROC_NAME
     END /* END PF1Ø */
WHEN VALUE(AIDKEY)='PF11' THEN DO                   /* SCROLL FORWARD */
     P=P+1; IF P>=PROCS.Ø THEN P=1
     UID.='        ';O_ADDR.='    ';T_ADDR.='    '
     MODE.='  ';LPSWD.='        ';A_MODE.='    ';UPDYN='N'
     SIGNAL LOAD_PROC_NAME
     END /* END PF11 */
```

```
WHEN VALUE(AIDKEY)='PFØ6' THEN DO                 /* DELETE PROCEDURE */
    'PIPE (ENDCHAR ?) STEM SCR_OUT. | A: FANOUT',
       '| TAKE FIRST | VAR N_PROC',
    '?A:| TAKE LAST  | VAR UPDYN '
  IF UPDYN¬='Y' THEN DO                            /* UPDATE FLAG OFF */
     'XMITMSG 61 (APPLID TAF CALLER XST NOCOMP VAR'
     PROC=N_PROC
     UID.='        ';O_ADDR.='    ';T_ADDR.='    '
     MODE.='  ';LPSWD.='        ';A_MODE.='    ';UPDYN='N'
     SIGNAL LOAD_ARRAYS
    END /* END UPDYN¬=Y */
              ELSE DO                              /* UPDATE FLAG ON  */
  IF STRIP(N_PROC)='' THEN DO                      /* PROCNAME NULL   */
  'XMITMSG 8Ø (APPLID TAF CALLER XST NOCOMP VAR'
   UPDYN='N';P=Ø
   SIGNAL XSTATMS
    END /* END PROCNAME NULL */
     IF FIND(PROCS,N_PROC)=Ø THEN DO           /* NON EXISTING PROC   */
     'XMITMSG 81 'N_PROC' (APPLID TAF CALLER XST NOCOMP VAR'
      P=Ø;PROC=N_PROC
      SIGNAL XSTATMS
      END /* END NON-EXISTING PROC */
    'PIPE < MENU XSTATMS ',
    '| NLOCATE (1.8) /'N_PROC'/',
    '| > MENU XSTATMS A'
    'XMITMSG 87 N_PROC (APPLID TAF CALLER XST NOCOMP VAR'
     SIGNAL RESET_ALL
    END /* END UPDYN=Y */
     END /* END PFØ6 */
WHEN VALUE(AIDKEY)='PF13' THEN DO                              /* TEST */
     SAY MENU
     SIGNAL XSTATMS
     END /* END PF13 TEST */
WHEN VALUE(AIDKEY)='PF24' | VALUE(AIDKEY)='PF12' THEN EXIT
WHEN VALUE(AIDKEY)='CLEAR' | VALUE(AIDKEY)='PA2' THEN SIGNAL RESET_ALL
OTHERWISE DO
     'XMITMSG 1 'VALUE(AIDKEY)' (APPLID TAF CALLER XST NOCOMP VAR'
     SIGNAL XSTATMS
     END /* END OTHERWISE */
END /* END SELECT */
RETURN
/*
*/
```

XSTATMS.MEN

```
================================================
         1         2         3         4
12345678901234567890123456789012345678901234567890
```

```
MENPROC==STATMS===PARMS====T_D==O_D==MD=LPSW====
Hxstatms OWNER    This is the procedure definition header
Hxstatms LINK     This is the definition  for CP LINK generation
Hxstatms LINK     Diskowner Disk_addr Virt_addr Mode Link_Password
Hxstatms LINK
Hxstatms ACCESS   This is the definition fo CMS ACCESS generation
Hxstatms ACCESS   Virt_addr Access_mode
Hxstatms ACCESS
Hxstatms ACCESS
Hxstatms TDISK    Temporary Disk specification: Y/N/Cyl_Number
Hxstatms EX       Focus program to execute
DICTI
DICTI    LINK     FOCUS    0065 0065 RR D3D3C1
DICTI    LINK     FOCUS    0223 0223 RR E2E4C3D6C6D9
DICTI    LINK     FOCUS    0224 0224 RR E2E4C3D6C6D9
DICTI    LINK     FOCUS    0225 0225 RR E2E4C3D6C6D9
DICTI    LINK     TESTDBS  0700 0700 RR E2C2C4D9
DICTI    ACCESS   0065 F
DICTI    ACCESS   0223 L/A
DICTI    ACCESS   0224 M/A
DICTI    ACCESS   0225 N/A
DICTI    ACCESS   0700 E
DICTI    TDISK    30
DICTI    EX       DICTION
SUPER
SUPER    LINK     FOCUS    0065 0065 RR D3D3C1
SUPER    LINK     FOCUS    0223 0223 RR E2E4C3D6C6D9
SUPER    LINK     FOCUS    0224 0224 RR E2E4C3D6C6D9
SUPER    LINK     FOCUS    0225 0225 RR E2E4C3D6C6D9
SUPER    LINK     FOCSUPER 0191 0005 RR D9C5D7E4E2D9
SUPER    ACCESS   0065 F
SUPER    ACCESS   0223 L/A
SUPER    ACCESS   0224 M/A
SUPER    ACCESS   0225 N/A
SUPER    ACCESS   0005 B
SUPER    TDISK    22
SUPER    EX       FOCSUPER
TESTE
TESTE    LINK     FOCUS    0065 0065 RR D3D3C1
TESTE    LINK     FOCUS    0223 0223 RR E2E4C3D6C6D9
TESTE    LINK     FOCUS    0401 0401 RR E2E4C3D6C6D9
TESTE    LINK     FOCUS    0405 0405 RR E2E4C3D6C6D9
TESTE    LINK     FOCUSTST 0401 0301 RR D3D3C1
TESTE    LINK     FOCUSTST 0402 0302 RR D3D3C1
TESTE    ACCESS   0065 F
TESTE    ACCESS   0223 L/A
TESTE    ACCESS   0401 M/A
```

```
TESTE    ACCESS   Ø4Ø5 N/A
TESTE    ACCESS   Ø3Ø1 B
TESTE    ACCESS   Ø3Ø2 C
TESTE    TDISK    Y
TESTE    EX       FTEST
FORM3
FORM3    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
FORM3    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
FORM3    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
FORM3    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
FORM3    LINK     FOCUS    Ø5ØØ Ø4Ø2 RR E2E4C3D6C6D9
FORM3    ACCESS   ØØ65 F
FORM3    ACCESS   Ø223 L/A
FORM3    ACCESS   Ø4Ø1 M/A
FORM3    ACCESS   Ø4Ø5 N/A
FORM3    ACCESS   Ø4Ø2 O/A
FORM3    TDISK    3
FORM3    EX       FORMS
EPICE
EPICE    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
EPICE    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
EPICE    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
EPICE    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
EPICE    LINK     EPICDB   Ø192 Ø192 RR D3D3C1
EPICE    ACCESS   ØØ65 F
EPICE    ACCESS   Ø223 L/A
EPICE    ACCESS   Ø4Ø1 M/A
EPICE    ACCESS   Ø4Ø5 N/A
EPICE    ACCESS   Ø192 B
EPICE    TDISK    Y
EPICE    EX       FEPIC
EPICP
EPICP    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
EPICP    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
EPICP    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
EPICP    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
EPICP    LINK     EPICDB   Ø195 Ø192 RR C4D9D7C3C9D7C5
EPICP    ACCESS   ØØ65 F
EPICP    ACCESS   Ø223 L/A
EPICP    ACCESS   Ø4Ø1 M/A
EPICP    ACCESS   Ø4Ø5 N/A
EPICP    ACCESS   Ø192 B
EPICP    TDISK    5Ø
EPICP    EX       PEPC
SASP
SASP     LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
SASP     LINK     SAS      Ø1ØØ ØØ66 RR D3D3C1
SASP     LINK     SASDB    Ø4Ø1 Ø4Ø1 RR E2C1E2D9
```

```
SASP     LINK     SASDB    Ø4Ø5 Ø4Ø5 RR E2C1E2D9
SASP     LINK     SASDB    Ø4ØA Ø4ØA RR E2C1E2D9
SASP     ACCESS   ØØ65 F
SASP     ACCESS   ØØ66 G
SASP     ACCESS   Ø4Ø1 M/A
SASP     ACCESS   Ø4Ø5 N/A
SASP     ACCESS   Ø4ØA O/A
SASP     TDISK    5Ø
SASP     EX       SAS
SASE
SASE     LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
SASE     LINK     SAS      Ø1ØØ ØØ66 RR D3D3C1
SASE     LINK     SASDB    Ø5Ø1 Ø5Ø1 RR E2C1E2D9
SASE     LINK     SASDB    Ø5Ø5 Ø5Ø5 RR E2C1E2D9
SASE     LINK     SASDB    Ø5ØA Ø5ØA RR E2C1E2D9
SASE     ACCESS   ØØ65 F
SASE     ACCESS   ØØ66 G
SASE     ACCESS   Ø5Ø1 M/A
SASE     ACCESS   Ø5Ø5 N/A
SASE     ACCESS   Ø5ØA O/A
SASE     TDISK    5Ø
SASE     EX       SAS
ACCNT
ACCNT    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
ACCNT    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
ACCNT    LINK     FCSACCNT Ø22Ø Ø22Ø RR E3D5C3C3C1D9
ACCNT    LINK     FCSACCNT Ø221 Ø221 RR E3D5C3C3C1D9
ACCNT    LINK     FOCUSTST Ø4Ø1 Ø222 RR D3D3C1
ACCNT    LINK     FOCUSTST Ø4Ø2 Ø224 RR D3D3C1
ACCNT    ACCESS   ØØ65 F
ACCNT    ACCESS   Ø223 G
ACCNT    ACCESS   Ø22Ø M/A
ACCNT    ACCESS   Ø221 N/A
ACCNT    ACCESS   Ø222 O
ACCNT    ACCESS   Ø224 P
ACCNT    TDISK    Y
ACCNT    EX       TACCNT
ACCNTP
ACCNTP   LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
ACCNTP   LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
ACCNTP   LINK     FCSACCNT Ø52Ø Ø22Ø RR D7E3D5C3C3C1D9
ACCNTP   LINK     FCSACCNT Ø521 Ø221 RR D7E3D5C3C3C1D9
ACCNTP   LINK     FOCUSPRD Ø4Ø1 Ø222 RR E2E4C3D6C6D9
ACCNTP   LINK     FOCUSPRD Ø4Ø2 Ø224 RR E2E4C3D6C6D9
ACCNTP   ACCESS   ØØ65 F
ACCNTP   ACCESS   Ø223 G
ACCNTP   ACCESS   Ø22Ø M/A
ACCNTP   ACCESS   Ø221 N/A
```

```
ACCNTP   ACCESS   Ø222 O
ACCNTP   ACCESS   Ø224 P
ACCNTP   TDISK    5Ø
ACCNTP   EX       PACCNT
MAFSUPER
MAFSUPER LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
MAFSUPER LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
MAFSUPER LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
MAFSUPER LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
MAFSUPER LINK     FOCUSTAG Ø4Ø1 Ø3Ø1 RR D3D3C1
MAFSUPER LINK     FOCUSTAG Ø4Ø2 Ø3Ø2 RR D3D3C1
MAFSUPER ACCESS   ØØ65 F
MAFSUPER ACCESS   Ø223 L/A
MAFSUPER ACCESS   Ø4Ø1 M/A
MAFSUPER ACCESS   Ø4Ø5 N/A
MAFSUPER ACCESS   Ø3Ø1 B
MAFSUPER ACCESS   Ø3Ø2 C
MAFSUPER TDISK    Y
MAFSUPER EX       GFOC
REPORTS
REPORTS  LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
REPORTS  LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
REPORTS  LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
REPORTS  LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
REPORTS  LINK     FOCUSPRD Ø4Ø1 Ø3Ø1 RR D3D3C1
REPORTS  LINK     FOCUSPRD Ø4Ø2 Ø3Ø2 RR D3D3C1
REPORTS  ACCESS   ØØ65 F
REPORTS  ACCESS   Ø223 L/A
REPORTS  ACCESS   Ø4Ø1 M/A
REPORTS  ACCESS   Ø4Ø5 N/A
REPORTS  ACCESS   Ø3Ø1 B
REPORTS  ACCESS   Ø3Ø2 C
REPORTS  TDISK    3Ø
REPORTS  EX       PREPS
REPSEX
REPSEX   LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
REPSEX   LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
REPSEX   LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
REPSEX   LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
REPSEX   LINK     REPUSER  Ø4Ø1 Ø191 MW E2D7C5D9E6
REPSEX   ACCESS   ØØ65 F
REPSEX   ACCESS   Ø223 L/A
REPSEX   ACCESS   Ø4Ø1 M/A
REPSEX   ACCESS   Ø4Ø5 N/A
REPSEX   ACCESS   Ø191 O/A
REPSEX   TDISK    Y
REPSEX   EX       REPORTS
PRODE
```

```
PRODE    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
PRODE    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
PRODE    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
PRODE    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
PRODE    LINK     FOCUSPRD Ø4Ø1 Ø3Ø1 RR D3D3C1
PRODE    LINK     FOCUSPRD Ø4Ø2 Ø3Ø2 RR D3D3C1
PRODE    ACCESS   ØØ65 F
PRODE    ACCESS   Ø223 L/A
PRODE    ACCESS   Ø4Ø1 M/A
PRODE    ACCESS   Ø4Ø5 N/A
PRODE    ACCESS   Ø3Ø1 B
PRODE    ACCESS   Ø3Ø2 C
PRODE    TDISK    3Ø
PRODE    EX       PFOC
```

ZLINES EXEC

```
 /* XLINES EXEC */
/* AIDKEY EQUATES */
ARG MENU
$F1='PFØ1';$F2='PFØ2';$F3='PFØ3';$F4='PFØ4';$F5='PFØ5';$F6='PFØ6'
$F7='PFØ7';$F8='PFØ8';$F9='PFØ9';$7A='PF1Ø';$7B='PF11';$7C='PF12'
$C1='PF13';$C2='PF14';$C3='PF15';$C4='PF16';$C5='PF17';$C6='PF18'
$C7='PF19';$C8='PF2Ø';$C9='PF21';$4A='PF22';$4B='PF23';$4C='PF24'
$Ø1='PA1';$6E='PA2';$7D='ENTER';$6D='CLEAR'
'SET LANGUAGE (ADD TAF USER'
'VMFCLEAR'
MESSAGE.1=COPIES(' ',56)

RESET_ALL:
MENLEXEC.=''
MENLDESC.=''
MENLTYPE.=''
UPDYN='N'
IF STRIP(MENU)='' THEN MENDESCR='                '
                  ELSE DO
'PIPE < MENU XLINES A ',
'| DROP 4',
'| LOCATE (1.9) /'MENU'/',
'| TAKE FIRST',
'| SPECS 2Ø.17 1 ',
'| VAR MENDESCR'
 IF MENDESCR='MENDESCR' THEN MENDESCR='                '
 ELSE 'XMITMSG 7Ø MENU (APPLID TAF CALLER ZLI NOCOMP VAR'
  END
LOAD_ARRAYS:
 ZLINES :
```

```
 SCREEN='8003'X||,
'1100002903C020410042F3'X||COPIES('=',77)||'11004E1DF0'X||,
'11004F2903C020410042F3'X||'|'||'1100511DF0'X||,
'1100632903C02041F242F5'X||' DYNAMIC MENUS ADMINISTRATION UTILITIES
'||'11008C1DF0'X||,
'11009D2903C020410042F3'X||'|'||'11009F1DF0'X||,
'11009F2903C020410042F3'X||'|'||COPIES('=',77)||'|'||'1100EF1DF0'X||,
'1100EF2903C020410042F3'X||'|'||'1100F11DF0'X||,
'11010A2903C02041F242F5'X||' MENUS CREATION UTILITY '||'1101231DF0'X||,
'11013D2903C020410042F3'X||'|'||'11013F1DF0'X||,
'11013F2903C020410042F3'X||'|'||COPIES('=',77)||'|'||'11018F1DF0'X||,
'11018F2903C020410042F3'X||'|'||'1101911DF0'X||,
'1101942903C02041F242F7'X||' MENU NAME : '||'1101A21DF0'X||,
'1101A52903C02041F442F7'X||MENU||'1101AE1DF0'X||,
'1101B02903C02041F242F7'X||' MENU DESCRIPTION : '||'1101C51DF0'X||,
'1101C82903C00141F442F7'X||MENDESCR||'1101DA1DF0'X||,
'1101DD2903C020410042F3'X||'|'||'1101DF1DF0'X||,
'1101DF2903C020410042F3'X||'|'||,
'11022D2903C020410042F3'X||'|'||'11022F1DF0'X||,
'11022F2903C020410042F3'X||'|'||COPIES('_',77)||'|'||'11027F1DF0'X||,
'11027F2903C020410042F3'X||'|'||'1102811DF0'X||,
'1102CD2903C020410042F3'X||'|'||'1102CF1DF0'X||,
'1102CF2903C020410042F3'X||'|'||'1102D11DF0'X||,
'1102D52903C02041F242F7'X||'LINE'||'1102DA1DF0'X||,
'1102DE2903C02041F242F7'X||'   DESCRIPTION'||'1102F01DF0'X||,
'1102F42903C02041F242F7'X||'  EXEC'||'1102FD1DF0'X||,
'1103012903C02041F242F7'X||'  TYPE'||'11030A1DF0'X||,
'11031D2903C020410042F3'X||'|'||'11031F1DF0'X||,
'11031F2903C020410042F3'X||'|'||'1103211DF0'X||,
'1103262903C030410042F4'X||'1'||'1103281DF0'X||,
'11032E2903C00141F442F4'X||MENLDESC.1 ||'1103401DF0'X||,
'1103442903C00141F442F4'X||MENLEXEC.1 ||'11034D1DF0'X||,
'1103512903C00141F442F4'X||MENLTYPE.1 ||'11035A1DF0'X||,
'11036D2903C020410042F3'X||'|'||'11036F1DF0'X||,
'11036F2903C020410042F3'X||'|'||'1103711DF0'X||,
'1103762903C030410042F4'X||'2'||'1103781DF0'X||,
'11037E2903C00141F442F4'X||MENLDESC.2 ||'1103901DF0'X||,
'1103942903C00141F442F4'X||MENLEXEC.2 ||'11039D1DF0'X||,
'1103A12903C00141F442F4'X||MENLTYPE.2 ||'1103AA1DF0'X||,
'1103BD2903C020410042F3'X||'|'||'1103BF1DF0'X||,
'1103BF2903C020410042F3'X||'|'||'1103C11DF0'X||,
'1103C62903C030410042F4'X||'3'||'1103C81DF0'X||,
'1103CE2903C00141F442F4'X||MENLDESC.3 ||'1103E01DF0'X||,
'1103E42903C00141F442F4'X||MENLEXEC.3 ||'1103ED1DF0'X||,
'1103F12903C00141F442F4'X||MENLTYPE.3 ||'1103FA1DF0'X||,
'11040D2903C020410042F3'X||'|'||'11040F1DF0'X||,
'11040F2903C020410042F3'X||'|'||'1104111DF0'X||,
'1104162903C030410042F4'X||'4'||'1104181DF0'X||,
```

```
'11041E2903C00141F442F4'X||MENLDESC.4 ||'1104301DF0'X||,
'1104342903C00141F442F4'X||MENLEXEC.4 ||'11043D1DF0'X||,
'1104412903C00141F442F4'X||MENLTYPE.4 ||'11044A1DF0'X||,
'11045D2903C020410042F3'X||'|'||'11045F1DF0'X||,
'11045F2903C020410042F3'X||'|'||'1104611DF0'X||,
'1104662903C030410042F4'X||'5'||'1104681DF0'X||,
'11046E2903C00141F442F4'X||MENLDESC.5 ||'1104801DF0'X||,
'1104842903C00141F442F4'X||MENLEXEC.5 ||'11048D1DF0'X||,
'1104912903C00141F442F4'X||MENLTYPE.5 ||'11049A1DF0'X||,
'1104AD2903C020410042F3'X||'|'||'1104AF1DF0'X||,
'1104AF2903C020410042F3'X||'|'||'1104B11DF0'X||,
'1104B62903C030410042F4'X||'6'||'1104B81DF0'X||,
'1104BE2903C00141F442F4'X||MENLDESC.6 ||'1104D01DF0'X||,
'1104D42903C00141F442F4'X||MENLEXEC.6 ||'1104DD1DF0'X||,
'1104E12903C00141F442F4'X||MENLTYPE.6 ||'1104EA1DF0'X||,
'1104FD2903C020410042F3'X||'|'||'1104FF1DF0'X||,
'1104FF2903C020410042F3'X||'|'||'1105011DF0'X||,
'1105062903C030410042F4'X||'7'||'1105081DF0'X||,
'11050E2903C00141F442F4'X||MENLDESC.7 ||'1105201DF0'X||,
'1105242903C00141F442F4'X||MENLEXEC.7 ||'11052D1DF0'X||,
'1105312903C00141F442F4'X||MENLTYPE.7 ||'11053A1DF0'X||,
'11054D2903C020410042F3'X||'|'||'11054F1DF0'X||,
'11054F2903C020410042F3'X||'|'||'1105511DF0'X||,
'1105562903C030410042F4'X||'8'||'1105581DF0'X||,
'11055E2903C00141F442F4'X||MENLDESC.8 ||'1105701DF0'X||,
'1105742903C00141F442F4'X||MENLEXEC.8 ||'11057D1DF0'X||,
'1105812903C00141F442F4'X||MENLTYPE.8 ||'11058A1DF0'X||,
'11059D2903C020410042F3'X||'|'||'11059F1DF0'X||,
'11059F2903C020410042F3'X||'|'||'1105A11DF0'X||,
'1105A62903C030410042F4'X||'9'||'1105A81DF0'X||,
'1105AE2903C00141F442F4'X||MENLDESC.9 ||'1105C01DF0'X||,
'1105C42903C00141F442F4'X||MENLEXEC.9 ||'1105CD1DF0'X||,
'1105D12903C00141F442F4'X||MENLTYPE.9 ||'1105DA1DF0'X||,
'1105ED2903C020410042F3'X||'|'||'1105EF1DF0'X||,
'1105EF2903C020410042F3'X||'|'||'1105F11DF0'X||,
'1105F52903C030410042F4'X||'10'||'1105F81DF0'X||,
'1105FE2903C00141F442F4'X||MENLDESC.10||'1106101DF0'X||,
'1106142903C00141F442F4'X||MENLEXEC.10||'11061D1DF0'X||,
'1106212903C00141F442F4'X||MENLTYPE.10||'11062A1DF0'X||,
'11063D2903C020410042F3'X||'|'||'11063F1DF0'X||,
'11063F2903C020410042F3'X||'|'||'1106411DF0'X||,
'11068D2903C020410042F3'X||'|'||'11068F1DF0'X||,
'11068F2903C020410042F3'X||'|'||COPIES('_',77)||'|'||'1106DF1DF0'X||,
'1106DF2903C020410042F3'X||'|'||'1106E11DF0'X||,
'1106E42903C02041F242F6'X||' ADD MENU ? '||'1106F11DF0'X||,
'1106F12903C00141F442F6'X||UPDYN||'1105F31DF0'X||,
'1106F32903C03041F142F2'X||MESSAGE.1||'11072C1DF0'X||,
'11072D2903C020410042F3'X||'|'||'11072F1DF0'X||,
```

```
'11Ø73Ø29Ø3C0Ø2Ø41Ø042F3'X||COPIES('=',77)||'11Ø77E1DFØ'X||,
'11Ø1C913'X
'PIPE (ENDCHAR ?) VAR SCREEN | FULLSCREEN ',
'| SPLIT AT ANYOF /'"11"X'/',
'| A: FANOUT',
'| TAKE FIRST',
'| SPECS /$/ 1 1-1 C2X NEXT',
'| VAR AIDKEY',
'? A:',
'| DROP FIRST',
'| SPECS 3-* 1',
'| STEM SCR_OUT.'
MESSAGE.1=COPIES(' ',5Ø)
SELECT
WHEN VALUE(AIDKEY)='ENTER' THEN DO              /* PROCESS ENTER KEY */
     'PIPE (ENDCHAR ?) STEM SCR_OUT.',
     '| A: FANOUT | TAKE FIRST | VAR MENDESCR',
     '?A: | TAKE LAST | VAR UPDYN ',
     '?A: | DROP 1 | DROP LAST | JOIN 2 /$/ | NLOCATE (1.1) /$/',
        '| NLOCATE  /$$/ | STEM FIELDS.'
      IF MENDESCR='MENDESCR' | MENDESCR='' THEN DO
      'XMITMSG 68 (APPLID TAF CALLER ZLI NOCOMP VAR'
      SIGNAL ZLINES
      END
      M=Ø
      DO N=1 TO FIELDS.Ø
       PARSE VAR FIELDS.N MENLDESC.N '$' MENLEXEC.N '$' MENLTYPE.N . . .
       IF POS((MENLTYPE.N),'DELETED MENU MENPROC EXEC REXX')=Ø THEN DO
      'XMITMSG 66 MENLTYPE.N (APPLID TAF CALLER ZLI NOCOMP VAR'
       SIGNAL ZLINES
         END
         ELSE DO
         M=M+1
         NLINE.M=LEFT(MENU,9)||LEFT(MENLEXEC.N,1Ø)||LEFT(MENLDESC.N,19),
               ||LEFT(MENLTYPE.N,9)
         END
      END
     NLINE.Ø=M
     IF UPDYN='Y' THEN DO
     HEADER=LEFT(MENU,9)||'KOTERET   '||LEFT(MENDESCR,19)||'TITLE'
    'PIPE LITERAL 'HEADER,
    '| APPEND STEM NLINE.',
    '| APPEND LITERAL 'LEFT(MENU,1Ø)||'***END***',
    '| > 'MENU' ZLINES A3'
    'PIPE < MENU XLINES A ',
    '| NLOCATE (1.8) /'MENU'/',
    '| APPEND < 'MENU' ZLINES A3',
    '| > MENU XLINES A'
```

```
    'XMITMSG 69 MENU (APPLID TAF CALLER ZLI NOCOMP VAR'
    UPDYN='N'
    SIGNAL ZLINES
     END
     ELSE DO
    'XMITMSG 61 MENU (APPLID TAF CALLER ZLI NOCOMP VAR'
     SIGNAL ZLINES
     END
    END /* END ENTER */
WHEN VALUE(AIDKEY)='PF13' THEN DO                              /* TEST */
     SAY MENU
     SIGNAL ZLINES
     RETURN
     END /* END PF13 TEST */
WHEN VALUE(AIDKEY)='PF24' | VALUE(AIDKEY)='PF12' THEN EXIT
WHEN VALUE(AIDKEY)='CLEAR' | VALUE(AIDKEY)='PA2' THEN SIGNAL RESET_ALL
OTHERWISE DO
     'XMITMSG 1 'VALUE(AIDKEY)' (APPLID TAF CALLER ZLI NOCOMP VAR'
     SIGNAL ZLINES
     END /* END OTHERWISE */
END /* END SELECT */
RETURN
/*
*/
```

Yaakov J Hazan
Technical Support Manager
Ynon Technologies & Computers Ltd (Israel)

A compressed dump/restore

The DDR utility has the option COMPACT to perform a compressed dump of a mini-disk to a tape device. A functionally similar dump/ restore facility may be required for which the dump device is actually a CMS disk file. This article gives a solution to this problem in the particular case of a CMS formatted mini-disk.

The two programs described here are VDUMP (the dump) and VREST (the restore). Both call the DASD Block I/O system service of CP (BLOCKIO), to process the disk to be dumped or restored. BLOCKIO presumes that the disk is formatted in blocks of equal size, which may be 512, 1024, 2048, or 4096 bytes long, so the disk must be CMS formatted. However, there are at least two benefits gained from the use of BLOCKIO, namely device-independent access to the virtual DASD and the opportunity to read or write a disk-block, referring to it by just its number.

The basic idea of the dump program is simple – read a disk block, compress it, and write the compressed string as a variable-length logical record into the dump file. To begin reading disk blocks, we have to tell BLOCKIO the block size, the offset, and the virtual device address of the DASD. Therefore, it is convenient to have the mini-disk accessed because, in this case, the blocksize could easily be extracted from the corresponding entry in the Active Disk Table.

The offset represents the number of sequential blocks used on the disk by the CMS file system to implement its structure. Because BLOCKIO bypasses the CMS file system, we always specify the offset as 0, gaining access to all blocks on the disk (not only the data blocks).

The virtual device address is extracted from the Device Table entry of the mini-disk.

After BLOCKIO receives the blocksize, the offset, and the virtual device address, it responds to the dump program with starting block number (which is always one in our case) and ending block number on the disk. In other words, BLOCKIO gives the disk size in blocks.

The first logical record that the dump program writes into the dump

file is composed of two words, containing the disk blocksize and disk size in blocks, respectively. It allows the restore program to compare these values with the disk blocksize, and disk size in blocks, of the disk to be restored. This checks that the target disk of the restore is correctly prepared.

When a disk block is read, it is passed to the module DMKDNC, the part of DDR that performs compression. If the compression is successful, DMKDNC returns the compacted string, preceded by a byte representing the number of the compaction table used, which can never have a value of zero. After that, DMKDNC ends with a zero return code. If the compression is not successful, DMKDNC ends with return code 1. In this case, the program VDUMP expands the unchanged disk block with a leading zero byte. Regardless of the success of the compression, the dump program writes the resulting string to the dump file as a logical record.

The actual flow of control is more complex, because, for performance reasons, I used multiple chained block I/O instead of single block I/O. For this reason I included several conditional assembly statements in the source. These will help the user in deciding how many buffers to use in a single I/O request to BLOCKIO, to achieve faster execution. The number of I/O buffers may be specified at assembly time by means of a SYSPARM parameter. If SYSPARM is not used, the default is eight buffers.

When generating the module VDUMP, MAINT's 194 mini-disk must be accessed, because this is where the TEXT files DMKDNC (compaction routine) and DMKDNT (compaction tables) reside. If this disk is accessed, the command LOAD VDUMP will load them automatically.

With the restore program, the flow of control resembles that described above, but is in the opposite direction. When VREST reads a logical record from the dump file, it submits it to the module DMKDDC (the decompaction part of DDR), which tries to decompress the record. If decompression is successful, the string produced is the next disk block to be written (restored). If the decompression is unsuccessful, DMKDDC ends with a return code of 2, which means that this string was not compressed before. In this case, VREST removes the leading

zero byte, and the remainder is the next block to be restored. After this, regardless of the success of the decompression, VREST calls BLOCKIO to perform the I/O operation.

The dump program is called using the command VDUMP V, where V is the filemode of the disk to be dumped. Similarly, the restore is called by the command VREST W, where W is the filemode of the disk to be restored. The dump file is specified to VDUMP and VREST by suitable FILEDEF commands. To help the user specify the required parameters, and to reduce the possibility of errors, I have written two EXECs (VDUMP and VREST) which, in the form of dialogue, fill the parameters, issue necessary commands, and call the programs VDUMP and VREST, respectively. These EXECs are called using their names. There are two further EXECs (DUMPV and RESTV) which EXECLOAD VDUMP and VREST, respectively, into virtual storage, and call them from there. This is necessary if, for example, VDUMP EXEC occasionally resides on the disk to be dumped.

Notes:

- These programs were developed and tested under VM/ESA Release 1.5 370 Feature on IBM 3380 DASD.
- The search order of MACLIBs used to assemble VDUMP and VREST is DMSGPI, DMSOM, OSMACRO, DMKGPI, DMKPSI, and DMKOM.

VDUMP ASSEMBLER

```
*        THE DUMP PROGRAM
         LCLC  &NUMBUF
         LCLA  &I,&E
         AIF   ('&SYSPARM' EQ '').OMITTED
&NUMBUF  SETC  '&SYSPARM'
         AGO   .GOON
.OMITTED ANOP
&NUMBUF  SETC  '8'
.GOON    ANOP
&E       SETA  &NUMBUF
         AIF ((&E LT 1) OR (&E GT 64)).OMITTED
         EJECT
         PRINT NOGEN
VDUMP    CSECT
```

```
NUMBUF   EQU   &NUMBUF
         USING VDUMP,R15
         STNSM SYSMASK,X'FE' DISABLE EXTERNAL INTERRUPTS
         STCTL C0,C0,CTRL0
         MVC   SAVCTRL0,CTRL0     SAVE CONTROL REGISTER 0 CONTENTS
         OI    CTRL0+3,X'02' TURN BIT 30 IN CTRL REG0 ON TO ENABLE IUCV
*                            EXTERNAL INTERRUPTS WHEN NEEDED
         LCTL  C0,C0,CTRL0
         DROP  R15
         EJECT
         STM   R14,R12,12(R13)
         LR    R12,R15
         USING VDUMP,R12     BASE REGISTER
         USING NUCON,R0
         USING DEVSECT,R6
         USING IPARML,R7
         USING ADTSECT,R8
         USING MULTBUFF,R11
         LA    R0,SAVE
         ST    R0,8(R13)
         ST    R13,SAVE+4
         LR    R13,R0
         CLI   8(R1),X'FF'   ANY ARGUMENTS?
         BNE   PRESENT       YES
         LA    R15,1000      NO
         ST    R15,RETCODE
         MVC   ERRORTXT,=CL80'DISK MODE NOT SPECIFIED'
         BAL   R3,WRTERM
         B     EXIT          RESTORE ORIGINAL CONDITIONS & REGS. EXIT
PRESENT  L     R8,IADT       ADDRESS OF THE FIRST ENTRY IN ADT
MODETEST CLC   ADTM,8(R1)    ACTIVE DISK TABLE ENTRY FOUND?
         BE    FOUND         YES
         ICM   R8,15,ADTPTR  NO. NULL ENTRY IN ACTIVE DISK TABLE?
         BNZ   MODETEST      NO. (R8)=ADDRESS OF THE NEXT ENTRY IN ADT
         LA    R15,1004      YES. DISK MODE INVALID
         ST    R15,RETCODE
         MVC   ERRORTXT,=CL80'DISK MODE INVALID'
         BAL   R3,WRTERM
         B     EXIT          RESTORE ORIGINAL CONDITIONS & REGS. EXIT
FOUND    L     R6,ADTDTA     (R6)=RELATED DEVICE TABLE ENTRY ADDRESS
         LH    R0,DEVADDR    VIRTUAL DISK ADDRESS
         STH   R0,VDEVADDR
         L     R0,ADTDBSIZ   (R0)=VIRTUAL DISK BLOCKSIZE
         ST    R0,BLKSIZE
         DROP  R6
         DROP  R8
         XC    PARMLIST,PARMLIST  CLEAR THE IUCV PARAMETER LIST
         LA    R7,PARMLIST   NOW R7 IS IUCV PARAMETER LIST BASE REG
DECLARE  IUCV  DCLBFR,PRMLIST=PARMLIST,BUFFER=INTPLIST,CONTROL=NO
         BZ    DCLBFROK      DECLARE BUFFER COMPLETED SUCCESSFULLY?
```

```
         BC    1,DCLBFR3     NO. CC=3 MEANS ERROR READING DIRECTORY
         BC    2,NEVER       CC=2. SHOULD NEVER OCCUR
         CLI   IPRCODE,X'13' IUCV EXT INTERRUPT BUFFER ALREADY EXISTS?
         BE    ABENDED       YES. MAY BE THE PROGRAM HAS ABENDED BEFORE
*                            THIS RUN
NEVER    MVC   ERRORTXT,=CL80'DECLARE BUFFER FAILED'  CC=2
*                            *BLOCKIO ERROR RET CODE STORED IN IPRCODE
         BAL   R3,WRTERM
         XR    R15,R15
         IC    R15,IPRCODE   PUT *BLOCKIO ERROR RETURN CODE IN R15
         ST    R15,RETCODE
         B     EXIT
DCLBFR3  MVC   ERRORTXT,=CL80'ERROR READING DIRECTORY'
         BAL   R3,WRTERM
         LA    R15,1008
         ST    R15,RETCODE
         B     EXIT
ABENDED  IUCV  RTRVBFR
         B     DECLARE
DCLBFROK XC    PARMLIST,PARMLIST  CLEAR THE PARAMETER LIST
         MVC   IPUSER(4),BLKSIZE       * FILL THE PARAMETER LIST WITH
         MVC   IPUSER+4(4),OFFSET      * PARMS NEEDED TO CONNECT TO
         MVC   IPUSER+8(2),VDEVADDR    * *BLOCKIO SYSTEM SERVICE
         MVC   IPVMID,=CL8'*BLOCKIO'
         MVC   IPMSGLIM,=X'00FF'  MAXIMUM 00255 OUTSTANDING MESSAGES TO
*                                 *BLOCKIO ALLOWED
        IUCV  CONNECT,CONTROL=NO,PRMDATA=YES,PRTY=NO,QUIESCE=NO,       *
               PRMLIST=PARMLIST
         BZ    CONNWAIT      CONNECT FUNCTION SUCCESSFULLY STARTED?
         MVC   ERRORTXT,=CL80'CONNECTION NOT ESTABLISHED'  NO. CC ¬= 0.
*                            *BLOCKIO ERROR RET CODE STORED IN IPRCODE
         BAL   R3,WRTERM
         XR    R15,R15
         IC    R15,IPRCODE   PUT *BLOCKIO ERROR RETURN CODE IN R15
         ST    R15,RETCODE
         B     RTRVBFR       TERMINATE USE OF IUCV
CONNWAIT HNDEXT SET,IUCVEXIT ACTIVATE IUCV EXTERNAL INTERRUPT HANDLER
         MVC   NEXTADDR,=AL3(CONNCMPL)
         LPSW  WAITPSW       ENABLE EXTERNAL INTERRUPTS AND WAIT FOR
*                            CONNECTION COMPLETE INTERRUPTION
CONNCMPL LA    R7,INTPLIST   NOW R7 IS IUCV INTERRUPT BUFFER BASE REG
         CLI   IPTYPE,X'02'  CONNECTION COMPLETED?
         BE    CONNOK        YES
         CLI   IPUSER,X'04'  NO, PATH SEVERED BY *BLOCKIO. CHECK IF THE
*                            CONNECTION WAS ESTABLISHED BEFORE THIS RUN
         BNE   CONNERR       IT WAS NOT ESTABLISHED. THIS IS AN ERROR
         XC    PARMLIST,PARMLIST  CONNECTION ALREADY ESTABLISHED. CLEAR
*                            IUCV PARAMETER LIST AND SEVER THE PATH
         LA    R7,PARMLIST   NOW R7 IS IUCV PARAMETER LIST BASE REG
         IUCV  SEVER,PRMLIST=PARMLIST,ALL=NO,PATHID=PATHID
```

```
         BZ    DECLARE       IF SEVER SUCCESSFUL TRY TO CONNECT AGAIN
         MVC   ERRORTXT,=CL8Ø'SEVER FAILED' CC=1
*                            IUCV ERROR RETURN CODE STORED IN IPRCODE
         BAL   R3,WRTERM
         XR    R15,R15
         IC    R15,IPRCODE   PUT IUCV ERROR RETURN CODE IN R15
         ST    R15,RETCODE
         B     RTRVBFR       TERMINATE IUCV USE AND EXIT
CONNERR  MVC   ERRORTXT,=CL8Ø'PATH SEVERED BY *BLOCKIO'
         BAL   R3,WRTERM
         XR    R15,R15
         IC    R15,IPUSER    PUT *BLOCKIO ERROR CODE IN R15
         ST    R15,RETCODE
         B     RTRVBFR       TERMINATE USE OF IUCV
CONNOK   MVC   PATHID,IPPATHID    SAVE THE PATH-ID
         MVC   STRBLOCK,IPUSER    START BLOCK RETURNED FROM *BLOCKIO
         MVC   ENDBLOCK,IPUSER+4  END BLOCK RETURNED FROM *BLOCKIO
         OPEN  (OUTDSK,(OUTPUT))  OPEN THE FILE TO RECEIVE THE DUMP
         L     R6,STRBLOCK
         L     R5,ENDBLOCK
         SR    R5,R6
         A     R5,=F'1'      (R5)=NUMBER OF BLOCKS MINI-DISK CONTAINS
         LA    RØ,12         PREPARE AND WRITE THE DUMP HEADER RECORD
         L     R2,=A(RDW)    (R2)=RECORD DESCRIPTOR WORD ADDRESS
         STH   RØ,Ø(R2)      LOGICAL RECORD LENGTH IS 12
         L     RØ,BLKSIZE    (RØ)=DISK BLOCK SIZE
         ST    RØ,4(R2)      PUT BLOCK SIZE INTO THE HEADER RECORD
         ST    R5,8(R2)      PUT NUMBER OF BLOCKS IN THE HEADER RECORD
         PUT   OUTDSK,(R2)   WRITE THE HEADER RECORD
         LA    R4,Ø
         D     R4,=A(NUMBUF) DIVIDE THE NUMBER OF BLOCKS THE MINI-DISK
*                            CONTAINS BY THE NUMBER OF DATA BUFFERS.
*                            (R4)=REMAINDER,(R5)=QUOTIENT
         LTR   R5,R5         IS THE QUOTIENT ZERO?
         BZ    RESTTEST      YES
QUOTLOOP LA    R7,PARMLIST   NOW R7 IS IUCV PARAMETER LIST BASE REG
         XC    PARMLIST,PARMLIST  CLEAR IUCV PARAMETER LIST
         LA    RØ,3
         ST    RØ,IPTRGCLS   MULTIPLE REQUEST TO *BLOCKIO
         LA    R1Ø,NUMBUF    NUMBER OF BUFFERS TO BE USED
         ST    R1Ø,IPRMMSG1  STORE IT IN PARMLIST
         LA    R11,MBUFLST   ADDRESS OF PLIST DEFINING THE BLOCKS TO BE
*                            PROCESSED AND THE BUFFERS TO BE USED
         ST    R11,IPRMMSG2  STORE IT IN PARMLIST
         L     R9,=A(DATABUFS)
FILLQUOT ST    R9,MBPBUFAD   STORE THE BUFFER ADDRESS IN MBUFLST
         ST    R6,MBPBKNUM   NUMBER OF THE BLOCK TO BE PROCESSED
         A     R9,BLKSIZE    (R9)=ADDRESS OF THE NEXT BUFFER
         A     R6,=F'1'      (R6)=NUMBER OF THE NEXT BLOCK
         LA    R11,MBUFLEN(R11)    (R11)=ADDRESS OF THE NEXT ENTRY IN
```

```
*                                MBUFLST
         BCT   R1Ø,FILLQUOT   FILL ALL ENTRIES IN MBUFLST
         IUCV  SEND,PRMLIST=PARMLIST,PATHID=PATHID,DATA=PRMMSG,PRTY=NO,*
               TYPE=2WAY
         BZ    SENDWAIT      NORMAL COMPLETION OF SEND REQUEST
NONZERO  MVC   ERRORTXT,=CL8Ø'NONZERO VALUE IN IPRCODE ON SEND REQUEST'
         BAL   R3,WRTERM
         XR    R15,R15       NONZERO VALUE STORED IN IPRCODE
         IC    R15,IPRCODE   PUT SEND ERROR RETURN CODE IN R15
         ST    R15,RETCODE
         B     RTRVBFR       TERMINATE USE OF IUCV
SENDWAIT MVC   NEXTADDR,=AL3(SENDCMPL)
         LPSW  WAITPSW       ENABLE EXTERNAL INTERRUPTS AND WAIT FOR
*                            IUCV MESSAGE COMPLETE INTERRUPTION
SENDCMPL LA    R7,INTPLIST   NOW R7 IS IUCV INTERRUPT BUFFER BASE REG
         CLI   IPTYPE,X'Ø7'  IUCV MESSAGE COMPLETE INTERRUPTION?
         BE    MESSCMPL      YES
         CLI   IPTYPE,X'Ø4'  PATH QUIESCED BY *BLOCKIO?
         BE    QUIESCED      YES
SEVERED  MVC   ERRORTXT,=CL8Ø'PATH SEVERED BY *BLOCKIO'    NO. SEVERED
         BAL   R3,WRTERM
         XR    R15,R15
         IC    R15,IPUSER    PUT *BLOCKIO ERROR CODE IN R15
         ST    R15,RETCODE
         B     RTRVBFR
QUIESCED MVC   ERRORTXT,=CL8Ø'PATH QUIESCED BY *BLOCKIO'
         BAL   R3,WRTERM
         XR    R15,R15
         IC    R15,IPUSER    PUT *BLOCKIO ERROR CODE IN R15
         ST    R15,RETCODE
         B     RTRVBFR
MESSCMPL XR    R15,15
         ICM   R15,15,IPRMMSG1    ANY ERRORS BEFORE I/O INITIATION?
         ST    R15,RETCODE
         BNZ   MESSERR       YES. ISSUE ERROR MESSAGE AND EXIT
         L     R9,=A(DATABUFS)
         LA    R11,MBUFLST
         LA    R1Ø,NUMBUF
PUTQLOOP XR    R15,R15
         ICM   R15,1,MBPSTAT ANY ERRORS DETECTED AFTER I/O INITIATION?
         ST    R15,RETCODE
         BNZ   MULTERR       YES. ISSUE ERROR MESSAGE AND EXIT
         ST    R9,INPSTRAD   NO. ADDRESS OF THE BLOCK TO BE COMPACTED
         LA    R1,CPLIST     ADDRESS OF THE COMPACTION ROUTINE'S PLIST
         L     R15,DMKDNCAD  (R15)=ADDRESS OF COMPACTION ROUTINE ENTRY
*                            POINT
         BALR  R14,R15       CALL THE COMPACTION ROUTINE
         ICM   R15,15,RETCODE     COMPACTION ROUTINE'S RETCODE IN R15
         BZ    COMPQOK       COMPACTION OK
         C     R15,=F'1'     COMPACTION RETURN CODE=1?
```

```
         BNE   COMPERR       NO. COMPACTION ERROR. SHOULD NOT OCCUR
         XC    RETCODE,RETCODE    YES. THIS IS NOT AN ERROR
         LR    R14,R9        BLOCK NOT COMPACTED. PREPARE FOR MVCL
         L     RØ,=A(OBUFFER)
         A     RØ,=F'1'      R14 AND RØ ARE SOURCE AND TARGET ADDRESSES
*                            FOR MVCL RESPECTEVELY
         L     R15,BLKSIZE   NUMBER OF BYTES TO BE MOVED FROM SOURCE
         LR    R1,R15        SAME TO TARGET
         MVCL  RØ,R14        MOVE THE UNCOMPACTED BLOCK TO OUTPUT AREA
         L     R15,=A(OBUFFER)
         MVI   Ø(R15),X'ØØ'  MARK THE RECORD AS UNCOMPACTED
         L     RØ,BLKSIZE
         A     RØ,=F'1'
         B     SKIPQCMP
COMPQOK  L     RØ,OUTSTRLN   (RØ)=LENGTH OF THE COMPACTED STRING
SKIPQCMP A     RØ,=F'4'      (RØ)=OUTPUT LOGICAL RECORD LENGTH
         L     R2,=A(RDW)    (R2)=ADDRESS OF OUTPUT LOGICAL RECORD AND
*                            ITS RECORD DESCRIPTOR WORD
         STH   RØ,Ø(R2)      FILL THE RECORD DESCRIPTOR WORD
         PUT   OUTDSK,(R2)   WRITE THE OUTPUT LOGICAL RECORD
         A     R9,BLKSIZE
         LA    R11,MBUFLEN(R11)
         BCT   R1Ø,PUTQLOOP
         BCT   R5,QUOTLOOP
         B     RESTTEST
MESSERR  MVC   ERRORTXT,=CL8Ø'*BLOCKIO ERROR BEFORE ANY I/O INITIATION'
         BAL   R3,WRTERM
         B     CLOSE
MULTERR  MVC   ERRORTXT,=CL8Ø'ERROR READING ONE OR MORE DISK BLOCKS'
         BAL   R3,WRTERM
         B     CLOSE
COMPERR  MVC   ERRORTXT,=CL8Ø'ERROR IN COMPACTION ROUTINE'
         BAL   R3,WRTERM
         B     CLOSE
RESTTEST LTR   R4,R4         IS THE REMAINDER ZERO?
         BZ    CLOSE         YES
         LA    R7,PARMLIST   NOW R7 IS IUCV PARAMETER LIST BASE REG
         XC    PARMLIST,PARMLIST  CLEAR IUCV PARAMETER LIST
         LA    RØ,3
         ST    RØ,IPTRGCLS   MULTIPLE REQUEST TO *BLOCKIO
         LR    R1Ø,R4        (R1Ø)=NUMBER OF DISK BLOCKS REMAINED
         ST    R1Ø,IPRMMSG1  STORE IT IN PARMLIST
         LA    R11,MBUFLST   ADDRESS OF PLIST DEFINING THE BLOCKS TO BE
*                            PROCESSED AND THE BUFFERS TO BE USED
         ST    R11,IPRMMSG2  STORE IT IN PARMLIST
         L     R9,=A(DATABUFS)
FILLREST ST    R9,MBPBUFAD   STORE THE BUFFER ADDRESS IN MBUFLST
         ST    R6,MBPBKNUM   NUMBER OF THE BLOCK TO BE PROCESSED
         A     R9,BLKSIZE    (R9)=ADDRESS OF THE NEXT BUFFER
         A     R6,=F'1'      (R6)=NUMBER OF THE NEXT BLOCK
```

```
         LA    R11,MBUFLEN(R11)    (R11)=ADDRESS OF THE NEXT ENTRY IN
*                                  MBUFLST
         BCT   R1Ø,FILLREST   FILL FIRST (R4) ENTRIES IN MBUFLST
         IUCV  SEND,PRMLIST=PARMLIST,PATHID=PATHID,DATA=PRMMSG,PRTY=NO,*
                   TYPE=2WAY
         BNZ   NONZERO       NON-ZERO VALUE STORED IN IPRCODE
         MVC   NEXTADDR,=AL3(CMPLSEND)
         LPSW  WAITPSW       ENABLE EXTERNAL INTERRUPTS AND WAIT FOR
*                            IUCV MESSAGE COMPLETE INTERRUPTION
CMPLSEND LA    R7,INTPLIST   NOW R7 IS IUCV INTERRUPT BUFFER BASE REG
         CLI   IPTYPE,X'Ø7'  IUCV MESSAGE COMPLETE INTERRUPTION?
         BE    CMPLMESS      YES
         CLI   IPTYPE,X'Ø4'  PATH QUIESCED BY *BLOCKIO?
         BE    QUIESCED      YES
         B     SEVERED       NO. SEVERED
CMPLMESS XR    R15,15
         ICM   R15,15,IPRMMSG1    ANY ERRORS BEFORE I/O INITIATION?
         ST    R15,RETCODE
         BNZ   MESSERR       YES. ISSUE ERROR MESSAGE AND EXIT
         L     R9,=A(DATABUFS)
         LA    R11,MBUFLST
         LR    R1Ø,R4
PUTRLOOP XR    R15,R15
         ICM   R15,1,MBPSTAT ANY ERRORS DETECTED AFTER I/O INITIATION?
         ST    R15,RETCODE
         BNZ   MULTERR       YES. ISUUE ERROR MESSAGE AND EXIT
         ST    R9,INPSTRAD   ADDRESS OF THE BLOCK TO BE COMPACTED
         LA    R1,CPLIST     ADDRESS OF THE COMPACTION ROUTINE'S PLIST
         L     R15,DMKDNCAD  (R15)=ADDRESS OF COMPACTION ROUTINE ENTRY
*                            POINT
         BALR  R14,R15       CALL THE COMPACTION ROUTINE
         ICM   R15,15,RETCODE     COMPACTION ROUTINE'S RETCODE IN R15
         BZ    COMPROK       COMPACTION OK
         C     R15,=F'1'     COMPACTION RETURN CODE=1?
         BNE   COMPERR       NO. COMPACTION ERROR. SHOULD NOT OCCUR
         XC    RETCODE,RETCODE    YES. THIS IS NOT AN ERROR
         LR    R14,R9        BLOCK NOT COMPACTED. PREPARE FOR MVCL
         L     RØ,=A(OBUFFER)
         A     RØ,=F'1'      R14 AND RØ ARE SOURCE AND TARGET ADDRESSES
*                            FOR MVCL RESPECTIVELY
         L     R15,BLKSIZE   NUMBER OF BYTES TO BE MOVED FROM SOURCE
         LR    R1,R15        SAME TO TARGET
         MVCL  RØ,R14        MOVE THE UNCOMPACTED BLOCK TO OUTPUT AREA
         L     R15,=A(OBUFFER)
         MVI   Ø(R15),X'ØØ'  MARK THE RECORD AS UNCOMPACTED
         L     RØ,BLKSIZE
         A     RØ,=F'1'
         B     SKIPRCMP
COMPROK  L     RØ,OUTSTRLN   (RØ)=LENGTH OF THE COMPACTED STRING
SKIPRCMP A     RØ,=F'4'      (RØ)=OUTPUT LOGICAL RECORD LENGTH
```

```
         L     R2,=A(RDW)    (R2)=ADDRESS OF OUTPUT LOGICAL RECORD AND
*                            ITS RECORD DESCRIPTOR WORD
         STH   RØ,Ø(R2)      FILL THE RECORD DESCRIPTOR WORD
         PUT   OUTDSK,(R2)   WRITE THE OUTPUT LOGICAL RECORD
         A     R9,BLKSIZE
         LA    R11,MBUFLEN(R11)
         BCT   R1Ø,PUTRLOOP
CLOSE    CLOSE OUTDSK        CLOSE THE DUMP FILE
RTRVBFR  IUCV  RTRVBFR       TERMINATE ALL USE OF IUCV
EXIT     HNDEXT CLR          DEACTIVATE IUCV EXTERNAL INTERRUPT HANDLER
         LCTL  CØ,CØ,SAVCTRLØ     RESTORE CONTROL REGISTER Ø CONTENTS
         STNSM WORK,X'ØØ'
         XR    R15,R15
         IC    R15,SYSMASK
         EX    R15,STOSM           RESTORE ORIGINAL SYSTEM MASK
         L     R15,RETCODE
         L     R13,SAVE+4
         L     R14,12(R13)
         LM    RØ,R12,2Ø(R13)
         BR    R14
         SPACE
IUCVEXIT STM   R14,R12,12(R13)    GENERAL EXTERNAL INTERRUPT EXIT
         LR    R15,R12
         USING IUCVEXIT,R15
         USING EXTUAREA,R1
         CLC   EXTUCODE,=X'4ØØØ'  IUCV EXTERNAL INTERRUPT?
         BNE   EXTRET             NO. IGNORE IT AND WAIT
         NI    EXTUPSW+1,255-X'Ø2'     TURN WAIT BIT IN PSW OFF
         NI    EXTUPSW,255-X'Ø1'  DISABLE EXTERNAL INTERRUPTS
EXTRET   LM    R14,R12,12(R13)    RESTORE REGISTERS
         BR    R14                EXIT
         DROP  R15
         DROP  R1
         SPACE
WRTERM   LINEWRT DATA=(ERRORTXT,L'ERRORTXT)
         BR    R3
         SPACE
STOSM    STOSM WORK,X'ØØ'
         SPACE
SAVE     DS    9D
PARMLIST DS    CL48     IUCV PARAMETER LIST   (DOUBLE WORD BOUNDARY!)
INTPLIST DS    CL48     IUCV INTERRUPT BUFFER (DOUBLE WORD BOUNDARY!)
WAITPSW  DS    D
         ORG   WAITPSW
         DC    X'FF06ØØØØØØ'
NEXTADDR DS    AL3
&I       SETA  1
MBUFLST  DS    ØD
 REPEAT  ANOP
         DC    X'Ø2'
```

```
         DS    X
         DC    H'Ø'
         DS    A
         DS    F
         DC    F'Ø'
&I       SETA  &I+1
         AIF   (&I LE &E).REPEAT
RETCODE  DS    F
SAVCTRLØ DS    F
CTRLØ    DS    F
BLKSIZE  DS    F
OFFSET   DC    F'Ø'
STRBLOCK DS    F
ENDBLOCK DS    F
VDEVADDR DS    H
PATHID   DS    H
WORK     DS    X
SYSMASK  DS    X
ERRORTXT DS    CL8Ø
*
OUTSTRLN DS    F
DMKDNCAD DC    V(DMKDNC)
DMKDNTAD DC    V(DMKDNT)
*
*              DMKDNC (COMPACTION ROUTINE) PARAMETER LIST FORMAT
*
*              R1 -> A(A(ENCODING TABLES))
*                    A(LENGTH OF INPUT STRING)
*                    A(INPUT STRING)
*                    A(LENGTH OF OUTPUT STRING)  (RETURNED BY DMKDNC)
*                    A(OUTPUT BUFFER)
*                    A(RETURN CODE)
*
CPLIST   DC    A(DMKDNTAD)
         DC    A(BLKSIZE)
INPSTRAD DS    A
         DC    A(OUTSTRLN)
         DC    A(OBUFFER)
         DC    A(RETCODE)
*
OUTDSK   DCB   DDNAME=OUTDD,DSORG=PS,MACRF=(PM),RECFM=VB,LRECL=41Ø1,   *
               BLKSIZE=32760
         LTORG
         SPACE
         ORG   VDUMP+4Ø96
&I       SETA  1
DATABUFS EQU   *
.LOOP    ANOP
         DS    CL4Ø96
&I       SETA  &I+1
```

```
         AIF   (&I LE &E).LOOP
         SPACE
OUTBUFF  DS    CL41Ø1
         ORG   OUTBUFF
RDW      DS    H
         DC    H'Ø'
OBUFFER  DS    CL4Ø97
         SPACE
         NUCON
         DEVSECT
         ADT
         COPY  IPARML
         EXTUAREA
         SPACE
MULTBUFF DSECT ,             MUST BE ON A DOUBLEWORD BOUNDARY
MBPRCOD  DS    X'Ø2'         REQUEST CODE: X'Ø2' - READ, X'Ø1' - WRITE
MBPSTAT  DS    X             BLOCK STATUS RETURNED BY *BLOCKIO
         DS    H'Ø'          NOT USED. MUST BE ZERO
MBPBUFAD DS    A             DATA BLOCK ADDRESS
MBPBKNUM DS    F             NUMBER OF THE BLOCK TO BE READ OR WRITTEN
         DS    F'Ø'          NOT USED. MUST BE ZERO
MBUFLEN  EQU   *-MULTBUFF
         SPACE
         REGEQU
         END   VDUMP
```

Editor's note: this article will be continued next month.

Vladimir Ivanov Valov
Chief Expert in System Programming
Information Centre of Ministry of Finance (Bulgaria)

Call for papers

Why not share your expertise and earn money at the same time? *VM Update* is looking for REXX EXECs, macros, program code, etc, that experienced VMers have written to make their life, or the lives of their users, easier. We will publish it (after vetting by our expert panel) and send you a cheque when the article is published. Articles can be of any length and can be sent or e-mailed to Robert Burgess at any of the addresses shown on page 2. Why not call now for a free copy of our *Notes for contributors*?

IBM has added two Sterling Software packages targeted at Web-enabling 3270 VM applications, and providing a graphical Web front-end to OfficeVision/VM.

VM:Webserver Gateway lets CGI scripts communicate with a VM server through the existing application client. It allows CGI scripts to drive full-screen applications and gain access to users' enterprise server resources, while keeping to existing application security structures. It also lets CGI scripts provide Web interfaces to all applications running on VM, as well as VSE, and OS/390.

VM:Webserver for OfficeVision, meanwhile, lets users interact with OfficeVision through a GUI that employs objects such as 3-D buttons, icons, drop-down selection lists, check boxes, radio buttons, and scroll bars. With it, users don't need to log-on to VM to use OfficeVision.

It also means that remote users don't need special software, such as terminal emulation programs, to access OfficeVision functions. With the GUI front end, OfficeVision Web pages can be modified with standard header and footer files. Also, no modifications to VM or OfficeVision are required, and no conversion or migration of existing OfficeVision information.

For further information contact:
Sterling Software, 1800 Alexander Bell Drive, Reston, VA 22091, USA.
Tel: (703) 264 8000.
Sterling Software, 1 Longwalk Road, Stockley Park, Uxbridge, Middlesex, UB11 1DB, UK.

* * *

Hitachi Data Systems has added a new capability for the synchronous Hitachi Remote Copy (HRC) feature, allowing it to transfer data between System/370 and System/390 processors running versions of the VM/ESA operating systems that don't support peer-to-peer remote copy.

Another new capability for HRC is said to make it simpler for users to access read-only data at the secondary location during suspension of data transfer operations.

For further information contact:
Hitachi Data Systems, PO Box 54996, 750 E Central Expressway, Santa Clara, CA 95056 0996, USA.
Tel: (408) 970 1000.

* * *

IBM has announced the year 2000-ready OfficeVision/VM Release 4. Among the new features will be an ability to cope with user A disks in VM/ESA's Shared File System, more CMS commands on OV/VM service machines, and a user exit for cursor selection of text in notes and documents.

The main point of the release, however, is that all OV/VM functions will now support dates from 1 January 1970 to 31 December 2069. The software's Database Manager, Mail Distribution Manager, and Mailbox Managers have been improved to allow the execution of more CMS commands on their consoles.

Also included are previously service-only functions, including the ability to let users change default options for the RECALL command, and accept more Internet/MIME mail into the OV/VM in-basket. The software runs on any VM/ESA platform.

For further information contact your local IBM representative.