149 MVS update

February 1999

In this issue

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.

Dataset I/O activity

INTRODUCTION

The DSIOSTAT program reads the type 42 SMF records (DFSMS statistics and configuration) subtype 6 (the dataset level I/O statistics). The records generated immediately after the recording of the type 30 interval records are not processed. Only the records generated at the CLOSE event are processed. The dataset access report is produced and the total number of processed SMF records is printed. For each dataset the following statistical data is displayed:

- Dataset name.
- Total number of I/Os.
- Time (in multiples of 128 micro-seconds):
  - average response time
  - average connect I/O time
  - average pending I/O time
  - average disconnect I/O time
  - average control unit queue time.
- Weight = (total number of I/Os) x (average response time).
- VOLSER.

The datasets are sorted by name. The SMF macro IGWSMF must be modified to obtain the appropriate mapping – the parameter &SMF42_06 must be changed from NO to YES. The following JCL is required to run the DSIOSTAT program:

```
//jobname  JOB (acc,in),'programmer',CLASS=A,MSGCLASS=T
//DSIOSTAT EXEC PGM=DSIOSTAT
//STEPLIB  DD DSN=your.loadlib,DISP=SHR
//SMFDATA  DD DSN=your.SMF.records,DISP=OLD
//PRINTOUT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SORTIN   DD UNIT=SYSDA,DISP=(NEW,PASS),SPACE=(TRK,(900,90))
//SORTOUT  DD UNIT=SYSDA,DISP=(NEW,PASS),SPACE=(TRK,(900,90))
//OUTPDDN  DD SYSOUT=K
```

OPERATIONAL ENVIRONMENT

The program was successfully tested in an MVS 5.2.2 (with DFSMS 1.3) and OS/390 Version 1 Release 3 environment. The procedure SUTIME was described in *MVS Update* Issue 102, March 1995. A copy of the date conversion subroutines, SUYYDDDF and SUDATE has not been supplied because most shops will have different requirements. Any date handling procedure can be used to perform the conversion of the date from TIME macro format to a specific format. Procedure IEFSD095 (block char routine) is a standard MVS procedure. When you link the block character subroutine concatenate the following standard MVS library to the //SYSLIB DD statement:

```
//     DD DSN=SYS1.AOSBØ,DISP=SHR  for IEFSDØ95 block character rtne
```

The program source code contains comments referring to previously published articles. These can be downloaded from the Xephon Web site if required.

DSIOSTAT

```
DSIOSTAT CSECT
         USING *,R1Ø,R11           ESTABLISH ADDRESSABILITY
         STM   R14,R12,12(R13)     SAVE3 REGISTERS
         LR    R1Ø,R15             SET FIRST BASE REGISTER
         LA    R11,2Ø48(R1Ø)       SET SECOND BASE REGISTER
         LA    R11,2Ø48(R11)       AND INCREMENT TO PROPER VALUE
         LR    R12,R13             STORE PREVIOUS SA ADDRESS
         LA    R13,SAVE3           LOAD ADDRESS OF THIS SAVE3 AREA
         ST    R12,SAVE3+4         CHAIN BACKWARDS
         ST    R13,8(R12)          CHAIN FORWARD
         LR    R8,R1
         GETMAIN R,LV=4ØØØØ
         LR    R9,R1 (R9) = ADDR. OF THE ALLOCATED VIRTUAL STORAGE AREA
         LTR   15,15
         BZ    OKGETMAT
         LA    R15,4
         B     FINI
OKGETMAT ST    R9,R9SAVE
         OPEN  (PRINTDCB,(OUTPUT))
         OPEN  (OUTPUDCB,(OUTPUT))
         LA    R7,Ø      (R7) = RETURN CODE
         BAL   R8,BLOCKPAG
         OPEN  (SORTLDCB,(OUTPUT))
         LA    R1,Ø
         ST    R1,R426TOTA
         ST    R1,INICFIRS     SET INDICATOR FOR THE FIRST RECORD
         OPEN  (SMFREDCB,(INPUT))
```

```
         USING SMF42RCL,R9
GETSMFRE EQU   *
         GET   SMFREDCB,SMF42RCL
         CLI   SMF42RTY,X'2A'  TEST IF RECORD TYPE 42
         BE    OKRECTYP
         B     NOTSMF42
OKRECTYP EQU   *  CHECK IF SUBTYPE 6; IF NOT -> DO NOT PROCESS
         LH    R1,SMF42STY
         LA    R3,6
         CR    R3,R1
         BE    PROCESS
         B     NOTSMF42
PROCESS  MVI   SORTPRIN,C' '
         MVC   SORTPRIN+1(L'SORTPRIN-1),SORTPRIN
         L     R3,R426TOTA
         A     R3,=F'1'        INCREASE COUNTER
         ST    R3,R426TOTA
         L     R3,INICFIRS
         XR    R1,R1
         CR    R1,R3
         BNE   NOTFIRST
         MVC   SMFDATE(4),SMF42DTE
         CALL  SUYYDDDF,(SMFDATE,STAMP),VL
         MVC   FIRSTDAT(12),STAMP
         MVC   TIME(4),SMF42TME
         CALL  SUTIME,(TIME,TIMESTAM),VL
         MVI   HH+2,C'H'
         MVI   MM+2,C'M'
         MVI   SS+2,C'S'
         MVC   FIRSTTIM(9),TIMESTAM
         XR    R1,R1
         LA    R1,1
         ST    R1,INICFIRS     SET INDICATOR FOR THE FIRST RECORD
NOTFIRST MVC   SMFDATE(4),SMF42DTE
         MVC   TIME(4),SMF42TME
         LR    R3,R9           (R3) = ADDRESS OF BEGIN OF THE RECORD
         LA    R1,36
         AR    R3,R1           (R3) = ADDRESS OF SMF42JHO
         USING SMF42JHO,R3
         XR    R1,R1
         LH    R1,SMF42JHN  (R1) = NUMBER OF JOB HEADER SECTIONS
         LR    R12,R1
         XR    R4,R4
         XR    R1,R1
         LH    R1,SMF42JHL  (R1) = LENGTH OF JOB HEADER SECTION
         LA    R4,Ø
         CR    R1,R4     LENGTH OF JOB HEADER SECTION = Ø?
         BE    GETSMFRE  YES, NO JOB HEADER SECTS, IGNORE THIS RECORD
         LA    R4,1
         CR    R12,R4    TEST IF ONE JOB HEADER SECTION
         BE    ONJHSECT  YES
         MVC   PRINT(133),BLANK
         MVC   PRINT+1(14),=C'SMF42JHN NE 1 '
```

```
         MVC   PRINT+2Ø(2Ø),=C'<<<<<<<<<<<<<<'
         PUT   PRINTDCB,PRINT
ONJHSECT EQU   *
         LR    R4,R9           (R4) = ADDRESS OF BEGIN OF THE RECORD
         L     R1,SMF42JHO     (R1) = OFFSET TO FIRST JOB HEADER SECT
         AR    R4,R1           (R4) = ADDRESS OF JOB HEADER SECTION
         USING S42JDJNM,R4     EST JOB HEADER SECTION ADDRESSABILITY
JOBHEABE EQU   *               START NEXT JOB HEADER SECTION
         CLI   S42JDCOD,X'Ø1'  TEST IF INTERVAL RECORD?
         BE    TES#JHSE        YES, IGNORE THIS JOB HEADER SECTION
         L     R1,S42JDDSO     (R1) = OFFSET TO FIRST DATASET HEADER
         XR    R7,R7
         CR    R1,R7     OFFSET TO 1ST DS HEADER SECTION = Ø?
         BE    TES#JHSE  YES, NO DS HEADER SECTION FOR THIS JOBH SECT
         XR    R1,R1
         LH    R1,S42JDDSL     (R1) = LENGTH OF FIRST DATA SET HEADER
         XR    R7,R7
         CR    R1,R7     LENGTH OF 1ST DS HEADER SECTION = Ø?
         BE    TES#JHSE  YES, NO DS HEADER SECTION FOR THIS JOBH SECT
         XR    R1,R1
         LH    R1,S42JDIOL     (R1) = LENGTH OF DATASET I/O SECTION
         XR    R7,R7
         CR    R1,R7     LENGTH OF DS I/O SECTION = Ø?
         BE    NOTPUTRE  YES, NO DS I/O SECTION FOR THIS DATASET
         L     R1,S42JDDSO     (R1) = OFFSET TO FIRST DATASET HEADER
STARTDSH EQU   *               START DATASET HEADER
         LR    R5,R9           (R5) = ADDRESS OF BEGIN OF THE RECORD
         AR    R5,R1           (R5) = FIRST DATASET HEADER
         USING S42DSNXT,R5     EST DATASET HEADER SECTION ADDRESS
         MVC   DSNAME(44),S42DSN
         MVC   VOLSER(6),S42DSVOL
         LR    R8,R9           (R8) = ADDRESS OF BEGIN OF THE RECORD
         L     R6,S42DSIOO     (R6) = OFFSET TO DATASET I/O STATS
         XR    R1,R1
         CR    R1,R6           S42DSIOO = Ø?
         BE    NOTPUTRE        YES
         AR    R8,R6           (R8) = DATASET I/O STATISTICS
         USING S42DSIOR,R8     EST DATASET I/O STATISTICS ADDRESS
         MVC   TOTALIO(4),S42DSION    TOTAL NUMBER OF I/OS
         MVC   AVRESPTI(4),S42DSIOR
         MVC   AVIOCONN(4),S42DSIOC
         MVC   AVIOPEND(4),S42DSIOP
         MVC   AVIODISC(4),S42DSIOD
         MVC   AVCUQUTI(4),S42DSIOQ
         L     R7,TOTALIO      (R7) = TOTAL I/O
         XR    R6,R6           (R6) = Ø
         XR    R1,R1
         L     R1,AVRESPTI
         MR    R6,R1
         ST    R7,WEIGHT
         CLC   DSNAME(4),=X'Ø4Ø4Ø4Ø4'
         BNE   OKDSNAME
         MVC   DSNAME(17),=C'DATA FROM THE JOB'
```

```
         MVC   DSNAME+18(8),S42JDJNM
OKDSNAME EQU   *
         PUT   SORTLDCB,SORTPRIN
NOTPUTRE EQU   *
         L     R1,S42DSNXT
         XR    R7,R7
         CR    R1,R7           TEST IF LAST DATASET FOR THE JOBHSECT?
         BE    TES#JHSE        YES,LAST
         L     R1,S42DSNXT     (R1) = OFFSET TO FIRST DATASET HEADER
         B     STARTDSH
TES#JHSE XR    R1,R1
         LH    R1,SMF42JHL
         AR    R4,R1  (R4) = ADDRESS OF THE NEXT JOB HEADER SECTION
         BCT   R12,JOBHEABE
NOTSMF42 B     GETSMFRE
ENDATA   CLOSE (SMFREDCB)
         CALL  SUYYDDDF,(SMFDATE,STAMP),VL
         MVC   LASTDATE(12),STAMP
         CALL  SUTIME,(TIME,TIMESTAM),VL
         MVI   HH+2,C'H'
         MVI   MM+2,C'M'
         MVI   SS+2,C'S'
         MVC   LASTTIME(9),TIMESTAM
         MVC   PRINT(133),BLANK
         MVI   PRINT,C'Ø'
         MVC   PRINT+1(24),=C'MVSUPDATE  19 JUNE 1998 '
         MVC   PRINT+25(16),=C'SMF RECORDS FROM'
         MVC   PRINT+42(3),FIRSTDAT
         MVC   PRINT+45(2),FIRSTDAT+3
         MVC   PRINT+48(3),FIRSTDAT+5
         MVC   PRINT+52(4),FIRSTDAT+8
         MVI   PRINT+56,C','
         MVC   PRINT+58(9),FIRSTTIM
         MVC   PRINT+68(2),=C'TO'
         MVC   PRINT+72(3),LASTDATE
         MVC   PRINT+76(2),LASTDATE+3
         MVC   PRINT+79(3),LASTDATE+5
         MVC   PRINT+83(4),LASTDATE+8
         MVI   PRINT+87,C','
         MVC   PRINT+88(9),LASTTIME
         MVC   PRINT+1ØØ(28),=C'MVS TOOLBOX: SMF42, 6, CLOSE'
         PUT   OUTPUDCB,PRINT
         MVC   PRINT(133),BLANK
         MVC   PRINT+1(39),=C'TOTAL NUMBER OF RECORDS TYPE 42 SUBTYPE'
         MVC   PRINT+4Ø(4),=C' 6 ='
         L     R1,R426TOTA
         BAL   R8,CONVEBOX
         MVC   PRINT+45(1Ø),RESULT
         PUT   PRINTDCB,PRINT
         CLOSE (SORTLDCB)
         LA    R1,PARMSORT             LOAD PARAMETER LIST
         LINK  EP=ICEMAN
         OPEN  (SORTOUT,(INPUT))
```

```
         XR    R3,R3
         LA    R3,5Ø       EST LINES PER PAGE NUMBER
         XR    R2,R2
         LA    R2,1        EST LINES PER PAGE COUNTER
         XR    R5,R5
         LA    R5,1
         BAL   R8,NEWPAGE
GETLOOP  GET   SORTOUT,SORTPRIN
         MVC   ODSNAME(44),DSNAME
         MVC   OVOLSER(6),VOLSER
         L     R1,TOTALIO
         BAL   R8,CONVEBOX
         MVC   OTOTALIO(1Ø),RESULT
         L     R1,AVRESPTI
         BAL   R8,CONVEBOX
         MVC   OAVRESPT(1Ø),RESULT
         L     R1,AVIOCONN
         BAL   R8,CONVEBOX
         MVC   OAVIOCON(1Ø),RESULT
         L     R1,AVIOPEND
         BAL   R8,CONVEBOX
         MVC   OAVIOPEN(1Ø),RESULT
         L     R1,AVIODISC
         BAL   R8,CONVEBOX
         MVC   OAVIODIS(1Ø),RESULT
         L     R1,AVCUQUTI
         BAL   R8,CONVEBOX
         MVC   OAVCUQUT(1Ø),RESULT
         L     R1,WEIGHT
         BAL   R8,CONVEBOX
         MVC   OTIOMARE(1Ø),RESULT
         MVC   PRINT(133),BLANK
         MVC   PRINT(133),OUTPUREC
         PUT   OUTPUDCB,PRINT
         CR    R2,R3           TEST IF PAGE IS FULL
         BNE   PAGENFUL        NOT
         XR    R2,R2
         BAL   R8,NEWPAGE
PAGENFUL AR    R2,R5
         B     GETLOOP
ENDODATA CLOSE (SORTOUT)
         MVI   PRINT,C'-'
         MVC   PRINT+1(L'PRINT-1),PRINT
         PUT   OUTPUDCB,PRINT
         BAL   R8,BLOCKPAG
         MVC   PRINT(133),BLANK
         MVI   PRINT,C'Ø'
         MVC   PRINT+1(27),=C'STEVE KOWALSKI 29 AUG 1998 '
         MVC   PRINT+1ØØ(28),=C'MVS TOOLBOX: SMF42, 6, CLOSE'
         PUT   OUTPUDCB,PRINT
         CLOSE (PRINTDCB)
         CLOSE (OUTPUDCB)
FINI     L     R13,4(R13)
```

```
         L     R9,R9SAVE
         FREEMAIN  R,LV=40000,A=(R9)
         LA    R7,0
         LR    R15,R7
         RETURN (14,12),RC=(15)
NEWPAGE  MVC   PRINT(133),BLANK
         MVI   PRINT,C'1'
         MVC   PRINT+2(13),=C'SMF EXTRACT  '
         MVC   PRINT+25(33),=C'DATASETS I/O STATISTICS  WEIGHT '
         MVC   PRINT+58(33),=C'= (TOTAL I/O) * (AVER RESP TIME) '
         MVC   PRINT+100(12),=C'REPORT DATE:'
         CALL  SUDATE,(DATE),VL
         MVC   PRINT+113(3),DATENAME
         MVC   PRINT+117(2),DATEDAY
         MVC   PRINT+120(3),DATEMONT
         MVC   PRINT+124(4),DATEYEAR
         PUT   OUTPUDCB,PRINT
         MVC   PRINT(133),BLANK
         PUT   OUTPUDCB,PRINT
         MVI   PRINT,C'-'
         MVC   PRINT+1(L'PRINT-1),PRINT
         MVI   PRINT,C' ‘
         PUT   OUTPUDCB,PRINT
         MVC   PRINT(133),BLANK
         MVC   PRINT+2(38),=C'DATASET NAME                          '
         MVC   PRINT+49(38),=C' TOTAL    — AVERAGE TIME IN MULTIPLES'
         MVC   PRINT+87(38),=C' OF 128 MICRO-SECONDS -     WEIGHT  V'
         MVC   PRINT+125(6),=C'OLSER ‘
         PUT   OUTPUDCB,PRINT
         MVC   PRINT(133),BLANK
         MVC   PRINT+49(38),=C' NUMBER   RESPONSE    CONNECT    PENDI'
         MVC   PRINT+87(38),=C'NG   DISCONNECT    CUNIT              '
         PUT   OUTPUDCB,PRINT
         MVC   PRINT(133),BLANK
         MVC   PRINT+49(38),=C' OF I/O                   I/O        I'
         MVC   PRINT+87(38),=C'/O          I/O    QUEUE               '
         PUT   OUTPUDCB,PRINT
         MVI   PRINT,C'-'
         MVC   PRINT+1(L'PRINT-1),PRINT
         MVI   PRINT,C' '
         PUT   OUTPUDCB,PRINT
         BR    R8
BLOCKPAG EQU   *
*     INSERT YOUR BLOCK PAGE CODE (EG AS PUBLISHED IN MVS UPDATE,
*     ISSUE 123, PAGE 20), IF REQUIRED.
         BR    R8
CONVEBOX EQU   *
         CVD   R1,PACKED
         MVC   COPYPATE(12),PATTERN
         ED    COPYPATE(12),PACKFIE2
         MVC   RESULT(10),COPYPATE+2
         BR    R8
PATTERN  DC    XL12'402020202020202020202120'
```

```
         DS    ØD
PACKED   DS    ØPL8
         DS    PL2
PACKFIE2 DS    PL6
COPYPATE DS    CL12
SAVE3    DS    18F
NUMBER   DS    F
R426TOTA DS    F
R9SAVE   DS    F
TIME     DS    F
RESULT   DS    CL1Ø
PRINT    DS    CL133
BLANK    DC    CL133' '
         DS    ØD
SORTPRIN DS    ØCL1ØØ
DSNAME   DS    CL44     DSNAME
TOTALIO  DS    F        TOTAL NUMBER OF I/O
AVRESPTI DS    F        AVERAGE RESPONSE TIME
AVIOCONN DS    F        AVERAGE I/O CONNECT TIME
AVIOPEND DS    F        AVERAGE I/O PENDING TIME
AVIODISC DS    F        AVERAGE I/O DISCONNECT TIME
AVCUQUTI DS    F        AVERAGE CONTROL UNIT QUEUE TIME
WEIGHT   DS    F        WEIGHT = TOTALIO * AVRESPTI
VOLSER   DS    CL6      VOLSER
         DS    CL23     FILLER
         DS    ØD
PARMSORT DC    X'8Ø',AL3(ADLST)
         CNOP  2,4
ADLST    DC    AL2(LISTEND-LISTBEG)
LISTBEG  DC    A(SORTA)     STARTING ADDRESS OF SORT STMT
         DC    A(SORTZ)     ENDING ADDRESS OF SORT STMT
         DC    A(RECA)      STARTING ADDRESS OF RECORD STMT
         DC    A(RECB)      ENDING ADDRESS OF RECORD STMT
         DC    A(Ø)         NO E15 EXIT
         DC    A(Ø)         NO E35 EXIT
LISTEND  EQU   *
SORTA    DC    C' SORT FIELDS=(1,44,CH,A)'  SORT BY DSNAME
*SORTA   DC    C' SORT FIELDS=(45,4,BI,D)'  SORT BY TOTAL I/O
SORTZ    DC    C' '
RECA     DC    C' RECORD TYPE=F,LENGTH=1ØØ '
RECB     DC    C' '
SMFDATE  DS    F
INICFIRS DS    F
FIRSTDAT DS    CL12
LASTDATE DS    CL12
FIRSTTIM DS    CL9
LASTTIME DS    CL9
DATE     DS    ØCL12
DATENAME DS    CL3
DATEDAY  DS    CL2
DATEMONT DS    CL3
DATEYEAR DS    CL4
```

```
STAMP    DS   ØCL12
DAY      DS   CL3      BLANK
DAYNO    DS   CL2      BLANK
MONTH    DS   CL3      BLANK
YEAR     DS   CL2      19
YEAR1    DS   CL2      BLANK
TIMESTAM DS   ØCL11
HH       DS   CL2      BLANK
         DS   CL1      BLANK
MM       DS   CL2      BLANK
         DS   CL1      BLANK
SS       DS   CL2      BLANK
         DS   CL1      BLANK
DD       DS   CL2      BLANK
PARMLIST DS    ØD
*     INSERT PARMLILIST FOR BLOCK PAGE CODE (EG AS PUBLISHED IN
*     MVS UPDATE, ISSUE 123, PAGE 2Ø), IF REQUIRED.
OUTPUREC DS    ØCL15Ø
         DS    CL1     CONTROL CHARACTER
ODSNAME  DS    CL44    DSNAME
         DS    CL1     FILLER
OTOTALIO DS    CL1Ø    TOTAL NUMBER OF I/O
         DS    CL1     FILLER
OAVRESPT DS    CL1Ø    AVERAGE RESPONSE TIME
         DS    CL1     FILLER
OAVIOCON DS    CL1Ø    AVERAGE I/O CONNECTING TIME
         DS    CL1     FILLER
OAVIOPEN DS    CL1Ø    AVERAGE I/O PENDING TIME
         DS    CL1     FILLER
OAVIODIS DS    CL1Ø    AVERAGE I/O DISCONNECT TIME
         DS    CL1     FILLER
OAVCUQUT DS    CL1Ø    AVERAGE CONTROL UNIT QUEUE TIME
         DS    CL1     FILLER
OTIOMARE DS    CL1Ø    WEIGHT = OTOTALIO * OAVRESPT
         DS    CL1     FILLER
OVOLSER  DS    CL6     VOLSER
         DS    CL21    FILLER
         DS    ØD
PRINTDCB DCB   MACRF=PT,RECFM=FBA,LRECL=133,DSORG=PS,DDNAME=PRINTOUT
OUTPUDCB DCB   MACRF=PT,RECFM=FBA,LRECL=133,DSORG=PS,DDNAME=OUTPDDN
SMFREDCB DCB   MACRF=GM,DSORG=PS,RECFM=VBS,DDNAME=SMFDATA,EODAD=ENDATA
SORTLDCB DCB   MACRF=PM,RECFM=FB,LRECL=1ØØ,DSORG=PS,DDNAME=SORTIN
SORTOUT  DCB   MACRF=GM,RECFM=FB,LRECL=1ØØ,DSORG=PS,                  *
               DDNAME=SORTOUT,EODAD=ENDODATA
         LTORG  LTORG LTORG LTORG LTORG LTORG LTORG LTORG
         IFASMFR  (42)
         END
```

Szczepan Kowlalski
Software Specialist
The Johannesburg Stock Exchange (South Africa)

Sample ISPF commands

INTRODUCTION

It is possible to add extra functionality for your TSO/ISPF by creating your own commands in a user command table. You can do this using ISPF option 3.9 or by using the ISPF command tool described in Issues 146 and 147 of *MVS Update*. Issue 146 of *MVS Update* also explains the ISPF command tables and how to customize ISPF to define a user command table.

Figure 1 shows some sample ISPF commands (as displayed by the ISPF commands tool). The following examples show various uses of commands:

- APPLID and SYSID only show an ISPF message. You could create a similar one to display the current USERID if that were useful.
- BR, DL, DV, ED, and EDREC use EXECs to invoke basic ISPF functions from the command line, without losing or stopping whatever else you were doing in ISPF.
- CMDS invokes a user-written application. It is the ISPF Commands tool detailed in *MVS Update* Issues 146 and 147.
- DTST invokes a standard ISPF function with a different PARM. This sample puts you into ISPF dialog test with the current applid, so you can display variables, tables, etc.
- LA invokes an undocumented ISPF function. This ISRDDN function was described in *MVS Update* Issue 144. Try it!
- HCD and ISMF invoke applications in the same way as from a selection panel. You should do the same with any application you would like to invoke from the command line. If you are installing/ testing something new – try invoking it like these samples.
- POFF, PON, and TSR execute basic TSO commands. These are good if you want to save typing: TSO tso_command_text.
- TSH enhances the use of a basic TSO command.

```
------------------------------ ISPF Commands --------------------------------
Option ==>
  1  Show only Command Descriptions                      _ Application id : ISR
  2  Display Command Table information                   / User table . . : MY
  3  Save your User Commands on disk                     _ Site table . . : SITE
                                                         _ System table . : ISP
  Command   Table  Description/Action
  ========  =====  ================================  Sorted by STANDARD ORDER
_   APPLID   MY   display ISPF Application-id
    ----     SELECT CMD(%APPLID)
_   BR       MY   recursive BROWSE
             SELECT CMD(%BRO &ZPARM) NEWAPPL(ISR)
_   CMDS     MY   customise my ISPF commands
    --       SELECT CMD(%ISPFCMDS)
_   DL       MY   dataset list (invokes ISPF 3.4)
    --       SELECT CMD(%DSLIST &ZPARM) NEWAPPL(ISR)
_   DTST     MY   ISPF dialog test (using &ZAPPLID)
    --       SELECT PGM(ISPYXDR) PARM(&ZAPPLID) NOCHECK SCRNAME(DTST)
_   DV       MY   list all datasets on volume (invokes ISPF 3.4)
    --       SELECT CMD(%DSLIST V &ZPARM) NEWAPPL(ISR)
_   ED       MY   recursive EDIT
             SELECT CMD(%EDI &ZPARM) NEWAPPL(ISR)
_   EDREC    MY   invoke EDIT RECOVERY
    ---      SELECT CMD(%EDREC) NEWAPPL(ISR)
_   HCD      MY   hardware configuration dialog
             SELECT CMD(%CBDCHCD &ZPARM) NEWAPPL(HCD)
_   IO       MY   recursive selection of ISPF primary panel
             SELECT CMD(%ISPFOPT &ZPARM) NEWAPPL(ISR)
_   ISMF     MY   ISMF
             SELECT CMD(%DGTFMD01 &ZPARM) NEWAPPL(DGT) NOCHECK
_   LA       MY   TSO dataset allocations (ISRDDN)
             SELECT PGM(ISRDDN)
_   POFF     MY   TSO PREFIX OFF
    ---      SELECT CMD(PROFILE NOPREFIX)
_   PON      MY   TSO PREFIX ON
    ---      SELECT CMD(PROFILE PREFIX(&ZUSER))
_   SYSID    MY   display System-id
    ---      SELECT CMD(%SYSID)
_   TSH      MY   display TSO HELP information
             SELECT CMD(%TSOHELP &ZPARM) NEWAPPL SCRNAME(TSOHELP)
_   TSR      MY   TSO RECEIVE
             SELECT CMD(RECEIVE)
```

Figure 1: Sample ISPF commands

Here are the EXECs used by the above commands:

APPLID EXEC

```
/*================================> REXX <================================*/
/*  APPLID:  Display the active ISPF Application id                       */
/*-----------------------------------------------------------------------*/
    Address ISPEXEC "VGET (ZAPPLID) ASIS"
    ZERRSM = "Appl-Id is" ZAPPLID                    /* short message */
    ZERRLM = "   The ISPF Application-id is "ZAPPLID /* long message  */
    ZERRALRM = "NO"                                  /*  no alarm     */
    Address ISPEXEC "SETMSG MSG(ISRZØØ2)"
    Return
```

BRO EXEC

```
/*======================== REXX ============================*/
/*  BRO - RECURSIVE BROWSE OF A DATASET                     */
/*                                                          */
/*        This is defined in an ISPF command table          */
/*         BR  : 'SELECT CMD(%BRO &ZPARM) NEWAPPL(ISR)'     */
/*                                                          */
/*        The user enters one of the following :            */
/*          a) BR             - basic BROWSE panel          */
/*          b) BR dsname      - BROWSE "dsname"             */
/*          c) BR dsname vol  - BROWSE "dsname" on "vol"    */
/*                                                          */
/* 98/Ø2/Ø1: The char ! is an alternative to ', for         */
/*           fully qualifying a dataset name.  This         */
/*           allows: IS 'BR !dsname!' from Info/Man.        */
/*==========================================================*/
 trace o
 Address ISPEXEC
"CONTROL ERRORS RETURN"
 Parse upper arg DSN VOL
 If DSN = '' Then "SELECT PGM(ISRBRO) PARM(ISRBROØ1)"
 Else Do
    DSN = Translate(DSN,"'","!")
   "BROWSE DATASET("DSN") VOLUME("VOL")"
    Select
       When RC = 12 Then Do
          ZERRSM = 'NO DATA TO BROWSE'
          ZERRLM = '    Empty sequential dataset or zero-length',
                   'member of a partitioned dataset'
          End
       When RC = 14 Then Do
          ZERRSM = 'MEMBER NOT FOUND'
          ZERRLM = '   The specified member was not found'
```

```
         End
      When RC = 16 Then Do
         ZERRSM = 'NO MEMBERS FOUND'
         ZERRLM = '  No members match the specified pattern',
                  'or no members in partitioned dataset'
         End
      When RC = 20 Then Do
         ZERRSM = 'DATASET NOT FOUND'
         ZERRLM = '   The dataset 'DSN' was not found (or some',
                  'other severe error)'
         End
      Otherwise
      End
   If ZERRSM <> 'ZERRSM' Then Do
      ZERRALRM = 'YES'
      ZERRHM   = 'ISR1B000'             /* standard BROWSE HELP pnl */
     "SETMSG MSG(ISRZ002)"
      End
   End
Exit
```

DSLIST EXEC

```
/*======================= REXX ============================*/
/*  DSLIST - DATASET LIST (same as option 3.4)             */
/*                                                         */
/*        This is defined in an ISPF command table         */
/*          DL : 'SELECT CMD(%DSLIST &ZPARM) NEWAPPL'      */
/*                                                         */
/*        The user enters one of the following :           */
/*          a) DL             - basic 3.4 panel            */
/*          b) DL dsmask      - list of "dsmask" d/sets    */
/*          c) DL V vol       - all d/sets on "vol"        */
/*          c) DL dsmask vol  - "dsmask" d/sets on "vol"   */
/*--CHANGES-------------------------------------------------*/
/*  - Updated for ISPF 4.x                                 */
/*  - Removing membername or relative GDG number from DSN */
/*=========================================================*/
  trace o
  address TSO
 "ISPEXEC CONTROL ERRORS RETURN"
  parse upper arg DSN VOL .            /* get invocation argument(s) */

  if DSN <> '' then do
    "ISPEXEC CONTROL NONDISPL ENTER" /* simulate ENTER on next panel */

     if VOL <> '' then do
        ZDLPVL = Left(VOL,6)      /* ensure volser only 6 bytes long */
        if DSN = 'V' then         /* show all datasets on the volume */
           ZDLDSNLV = ''          /* ..therefore a blank dsname mask */
        end
```

```
     else ZDLPVL = ''
    "ISPEXEC VPUT ZDLPVL"                  /* VPUT the volume serial */

     if DSN <> 'V' then do
        if left(DSN,1) = "'" then
           DSN = strip(DSN,,"'")            /* strip quotes from DSN */
        else if SYSVAR(SYSPREF) <> '' then
           DSN = SYSVAR(SYSPREF)||'.'||DSN  /* add TSO prefix to DSN */
        mempos = pos('(',DSN)
        if mempos > Ø then   /* remove member or relative GDG number */
           DSN = left(DSN,mempos-1)
        ZDLDSNLV = DSN
        end

    "ISPEXEC VPUT ZDLDSNLV"                  /* VPUT the dsname mask */
     end
  "ISPEXEC SELECT PGM(ISRUDL) PARM(ISRUDLP)"    /* invoke normal 3.4 */
   exit
```

EDI EXEC

```
/*============================== REXX ================================*/
/*  EDI - RECURSIVE EDIT OF A DATASET                                 */
/*                                                                    */
/*     This could be defined as an ISPF command in an ISPF command    */
/*     table (in the ISPTLIB concatenation) like:                     */
/*                                                                    */
/*      ED1 : "SELECT CMD(%EDI 'dsname' PANEL('pnl') MACRO('mac')"    */
/*        to always edit the same 'dsname' using the panel and macro  */
/*                                                                    */
/*      ED  : "SELECT CMD(%EDI &ZPARM) NEWAPPL(ISR)"                  */
/*        The user enters parameter(s) like the following:            */
/*          a) ED             - basic EDIT panel                    */
/*          b) ED dsname      - EDIT "dsname"                       */
/*          c) ED dsname vol  - EDIT "dsname" on volume "vol"        */
/*          d) ED dsname parm - EDIT "dsname" with EDIT parameters    */
/*====================================================================*/
 trace o
 address ISPEXEC
"CONTROL ERRORS RETURN"
 parse upper arg DSN parm
 If DSN = '' then "SELECT PGM(ISREDIT) PARM(P,ISREDMØ1)"
 Else do
    If Length(parm) = 6 & Pos('(',parm) = Ø Then
       "EDIT DATASET("DSN") VOLUME("parm")"
    Else
       "EDIT DATASET("DSN")" parm
    Select
       when RC = 14 then do
          ZERRSM = 'DATA IN USE'
          ZERRLM = '    The specified member is already being editted'
```

```
         end
       when RC = 16 then do
          ZERRSM = 'MEMBER(S) NOT FOUND'
          ZERRLM = ' No members found matching the specified pattern,',
                   'or there are no members in the dataset'
          end
       when RC = 20 then do
          ZERRSM = 'SEVERE ERROR'
          ZERRLM = '   The dataset 'DSN' was not found, or some',
                   'other severe error'
          end
       otherwise
       end
    If ZERRSM <> 'ZERRSM' then do
       ZERRALRM = 'YES'
       ZERRHM   = 'ISR20000'              /* standard EDIT HELP pnl */
      "SETMSG MSG(ISRZ002)"               /* standard IBM dummy msg */
       end
    end
 Exit
```

EDREC EXEC

```
/*========================= REXX ==============================*/
/*  EDREC - PROCESS EDIT RECOVERY (if it is pending)           */
/*                                                             */
/*        This is defined in an ISPF command table             */
/*          EDREC : 'SELECT CMD(%EDREC) NEWAPPL(ISR)'          */
/*============================================================*/
   trace o
   address ISPEXEC
  "CONTROL ERRORS RETURN"
   do query = 1 to 8            /* max of 8 recoveries can be pending */
      ZERRALRM = 'YES'                        /* alarm with message */
      ZERRHM = 'ISR2J000'          /* standard HELP for EDIT RECOVERY */
     "EDREC QUERY"
      if RC <> 4 then do
         if query = 1 then do
            ZERRSM = 'NO RECOVERY PENDING'
            ZERRLM = ' '
           "SETMSG MSG(ISRZ002)"
            end
         leave query
         end
      ZERRSM = ''
      ZERRLM = "  *** EDIT RECOVERY - enter 'CAN' to cancel, or 'PF3'",
                "to save this version ***"
     "SETMSG MSG(ISRZ002)"
     "EDREC PROCESS"
      end
   return
```

ISPFOPT EXEC

```
/*============================ REXX ================================*/
/* ISPFOPT: Select an ISPF primary option from anywhere in ISPF.    */
/*          This appears "over the top" of the current active ISPF  */
/*          application (on the same side of any split screen, and  */
/*          with the same &ZSCREEN number).                         */
/*                                                                  */
/*          This should be set up in an ISPF command table as:      */
/*          IO   'SELECT CMD(%ISPFOPT &ZPARM) NEWAPPL(ISR)'         */
/*                                                                  */
/*          Examples of how this is used:                           */
/*          User enters: IO      ->  display a primary option panel */
/*                       IO 6    ->  invoke primary option 6        */
/*                       IO 3.4  ->  invoke primary option 3.4      */
/*                  Note: =X     ->  return to the original panel   */
/*==================================================================*/

trace O
 Address ISPEXEC           /* more efficient to 'Address' only once */
'CONTROL DISPLAY SAVE'                      /* save current display */

 Parse Arg option                     /* get user-specified option */

'VGET ZTSICMD'                       /* get ISPF invocation command */
 Parse Upper Var ZTSICMD . 'PANEL('panel')' .
 If panel = '' Then panel = 'ISR1PRIM'    /* set default panel used */

'SELECT PANEL('panel') OPT('option')'

'CONTROL DISPLAY RESTORE'                    /* restore old display */
 Return
```

SYSID EXEC

```
/*======================== REXX ================================*/
/*  SYSID - DISPLAYING THE MVS SYSTEM-ID                        */
/*==============================================================*/

   If Sysvar(SYSISPF) = 'ACTIVE' Then Do
      Address ISPEXEC "VGET ZSYSID"
      ZEDSMSG = "This system is "ZSYSID
      Address ISPEXEC "SETMSG MSG(ISRZ000)" /* IBM edit macro message */
      End
   Else Do
      smfid = ,
      Storage(D2X(C2D(Storage((D2X(C2D(Storage(4C,4))+197)),3))+16),4)
      Say " ***  This system is "smfid
      End
   Return
```

TSOHELP EXEC

```
/*====================== REXX ============================*/
/*  TSOHELP - DISPLAY TSO HELP IN ISPF BROWSE             */
/*        This is defined as an ISPF command:             */
/*          TSOHELP : 'SELECT CMD(%TSOHELP &ZPARM)'       */
/*        The user enters one of the following :          */
/*          a) TSOHELP         - basic HELP information   */
/*          b) TSOHELP cmdname - HELP for 'cmdname'       */
/*        The TSO line output is put into a dataset       */
/*        called 'userid.TSO.HELP' then browsed using     */
/*        ISPF BROWSE. The dataset is kept so that the    */
/*        user can print it if they wish.                 */
/*========================================================*/
 Trace O
 Parse Upper Arg cmd                  /* get the HELP argument      */
 x = OutTrap('HELP_Output.',5000)     /* up to 5000 lines of output */
 Address TSO "HELP" cmd               /* issue TSO HELP command     */
 x = OutTrap('Off')
 zuser = Userid()
 If Sysvar(SYSPREF) <> zuser Then
    zuser = Sysvar(SYSPREF)||'.'||zuser
 outds = "'"zuser".TSO.HELP'"         /* output dataset name        */
 If Sysdsn(outds) = "OK"
    Then alloc_info = "SHR REUSE"
    Else alloc_info = "NEW UNIT(3390) SPACE(1 1) TRACKS RECFM(V B)",
                      "LRECL(125) DSORG(PS) REUSE"
 "ALLOC F(TSOHELP) DATASET("outds")" alloc_info
 "EXECIO * DISKW TSOHELP (STEM HELP_Output. OPEN FINIS "
 Address ISPEXEC "ISPEXEC BROWSE DATASET("outds")"
 Return
```

WARNING

Remember that your ISPF commands should have names that are *not* the same as a selection panel option, or a command used by any ISPF application that you run, because ISPF will execute your command's action from your ISPF command table instead of passing the characters that you entered to the selection panel or the application. If you have installed the ISPF commands tool you can simply invoke it from any application (use the CMDS command above) and change any command(s) immediately if you need to. You might rename an offending command or temporarily delete one.

To conclude I would strongly recommend that you try using some extra ISPF commands because they can be a simple but powerful enhancement for your activities in ISPF.

Ron Brown
Systems Programmer (Germany)

Condition code checking for non-abending programs

INTRODUCTION

In *MVS Update* Issue 49 (October 1990), I published my original condition code checking utility, which would invoke a user-specified program, check its condition code upon return, and would abend the job step if a non-zero return code was returned by the invoked program. Since then, I have added several features to increase the flexiblity of the program. It can now specify what return code is acceptable – for example, a return code of four or less with IDCAMS will not cause an abend to be issued. It also has a program name table, which contains the names of programs that require some special handling (such as BMC utilities) in the form of using the MVS LINK macro rather than ATTACH to invoke them; this was required because BMC utilities check to see if they are ATTACHed, which they do not allow themselves to be.

THE PROBLEM

There are some utilities, both from IBM and other vendors, which try to be very forgiving when it comes to terminating their execution when errors are encountered. One of the most common is IDCAMS, the Access Methods Services utility. It tends to return a non-zero condition code rather than abend a step upon encountering an error. It does not even have any option to allow it to abend instead of returning non-zero condition codes.

This philosophy can present a large problem in batch production jobs and to some job scheduling systems. Unless condition code checking is performed on each step following a step returning a non-zero condition code, the subsequent steps will be executed, even if not desired. We ran into such a problem, again most noticeably with IDCAMS, in our batch production environment. We also have no standard which requires the use of condition code checking in production JCL, and our job scheduling system does not react to non-zero condition codes until the end of a job.

A SOLUTION

Rather than go into all production JCL to add condition code checking, we tried to find another quick method to fix the problem. While condition code checking could be done at the job level, that would have required massive manual updating to add the required parameter to each batch production JOB card. What we opted for is a front-end program that will issue an MVS LINK to the desired program (IDCAMS for example) and examine the return code passed by the called program. If the called program returns a non-zero condition code, the driver program issues an MVS ABEND. We decided to leave the task of determining the proper condition codes to our applications programmers. This is because utilities like IDCAMS can return the same condition codes for an acceptable event (like deleting a dataset which does not exist or is currently in use).

We wrote a driver program, called CONDCODE, to replace the program specified on the EXEC statement. The actual program to be called is passed as a parameter to the CONDCODE program along with any parameters to be passed to the actual program. The syntax of the EXEC statement is similar to that of the MVS LOADER program – the program name to be executed is coded first and if any PARM field is to be passed to that program, a slash (/) is coded following the program name, after which the parameter data to be passed to the program is coded. The sample JCL below provides an example using the IFCEREP1 program.

```
//SYSEXN   EXEC PGM=CONDCODE,REGION=10M,PARM='IFCEREP1/CARD'
//ACCIN    DD DISP=OLD,DSN=P.LOGREC.DAILY
//DIRECTWK DD UNIT=SYSDA,SPACE=(CYL,(50,15))
//TOURIST  DD SYSOUT=*,DCB=BLKSIZE=133
//EREPPT   DD SYSOUT=*,DCB=BLKSIZE=133
//SYSIN    DD *
SYSEXN=Y,ACC=N,HIST=Y,TABSIZE=999K
ENDPARM
LIMIT 3490,HR3490=025(1),HW3490=999(1),VR3490=025(1),VW3490=999(1)
LIMIT 3480,HR3480=025(1),HW3480=999(1),VR3480=025(1),VW3480=999(1)
/*
//CONDCODE DD *
MAX-ALLOWABLE-COND-CODE=0004
/*
```

The PARM data specifies that the program to be invoked is IFCEREP1; the data to the right of the slash (/) specifies that the parameter CARD is to be passed to the IFCEREP1 program. If IFCEREP1 were to return

a condition code greater than four to the CONDCODE program, it would issue a user ABEND to prevent the execution of any further steps in a job. This is specified by including the optional CONDCODE DD statement which contains a single control card. The syntax of the control card, starting in column one is:

```
MAX-ALLOWABLE-COND-CODE=xxxx
```

where xxxx is a 4-byte numeric value specifying the maximum allowable condition code that the CONDCODE program will deem acceptable. This is how utilities, such as IDCAMS mentioned earlier, can return non-zero condition code values that will be accepted without CONDCODE issuing an abend. In its simplest implementation, we use a global update facility to replace all occurrences of:

```
EXEC PGM=IDCAMS   with
EXEC PGM=CONDCODE,PARM=IDCAMS
```

This allows us to terminate any job in which IDCAMS returns a non-zero condition code with a minimum of interruption to our JCL decks. We have identified other utilities which return non-zero condition codes rather than abend; for that we have implemented the above method for condition code checking. This is the quickest route for us to ensure that subsequent steps of a job do not execute, and it is much faster to implement than going into these job streams to add condition code checking. If more specific condition code checking is required, then manual editing of the job streams would need to be done to add the CONDCODE DD statement and its required control card.

CONDCODE

```
CONDCODE TITLE 'CONDCODE - DRIVER PROGRAM TO PERFORM CONDITION CODE CHE+
               CKING'
*  CONDCODE CAN ONLY BE USED TO CALL PROGRAMS WHICH RESIDE IN APF
*  AUTHORIZED LIBRARIES, BUT THE PROGRAM CALLED DOES NOT NEED AC(1)
CONDCODE CSECT                      ESTABLISH CSECT
CONDCODE AMODE 24
CONDCODE RMODE 24
         PRINT NOGEN
         SAVE  (14,12),,CONDCODE-&SYSDATE
         YREGS
         LR    R12,R15             LOAD A(EPA)
         USING CONDCODE,R12        ESTABLISH ADDRESSABLITY TO CSECT
         LA    R8,SAVEAREA         LOAD ADDR OF MY S/A
```

```
         ST    R8,8(,R13)          ST MY S/A ADDR IN CALLERS S/A
         ST    R13,4(,R8)          ST CALLERS S/A ADDR IN MY S/A
         LR    R13,R8              LOAD ADDR OF MY S/A IN R13
         L     R1,0(,R1)           LOAD A(PARM LIST)
         LH    R2,0(,R1)           LOAD PARM LIST LENGTH
         LTR   R2,R2               IS THERE A PARM FIELD GIVEN
         BNZ   PROCESS             YES, GO PROCESS IT
         WTO   'CCD001I NO PROGRAM NAME SPECIFIED IN PARM FIELD-ABENDINX
               G',ROUTCDE=11
         ABEND 0001,DUMP,,USER     ABEND FOR MISSING PROGRAM NAME
PROCESS  CH    R2,=H'100'          IS PARM LENGTH GREATER THAN MAX
         BNH   STORELEN            NO, GO SAVE PARM LENGTH
         WTO   'CCD002I PARAMETER LENGTH EXCEEDS MAXIMUM OF 100 CHARACTX
               ERS-ABENDING',ROUTCDE=11
         ABEND 0002,DUMP,,USER     ABEND FOR BAD PROGRAM NAME
STORELEN STH   R2,PARMLEN          STORE PARM FIELD LENGTH
         LA    R3,2(,R1)           LOAD A(ACTUAL PARM DATA)
         LA    R6,9                LOAD NUMBER OF TIMES TO LOOP
         LR    R8,R3               SAVE A(PARM DATA)
         SR    R4,R4               CLEAR FOR COUNTER
PGMLOOP  CLI   0(R3),C'/'          DID WE REACH END OF PGM NAME
         BE    RESETPRM            YES, GO FIX PARM FIELD
         CLI   0(R3),C'@'          NO, IS IT A VALID CHARACTER
         BE    BUMP                YES, GO BUMP TO NEXT BYTE
         CLI   0(R3),C'$'          IS IT A VALID CHARACTER
         BE    BUMP                YES, GO BUMP TO NEXT BYTE
         CLI   0(R3),C'#'          IS IT A VALID CHARACTER
         BE    BUMP                YES, GO BUMP TO NEXT BYTE
         CLI   0(R3),C'A'          IS IT A VALID CHARACTER
         BNL   BUMP                YES, GO BUMP TO NEXT CHARACTER
ABEND4   WTO   'CCD003I PROGRAM NAME IN PARM FIELD CONTAINS INVALID CHAX
               RACTERS-ABENDING',ROUTCDE=11
         ABEND 0003,DUMP,,USER     ABEND FOR BAD PROGRAM NAME
BUMP     LA    R3,1(,R3)           BUMP TO NEXT BYTE
         LA    R4,1(,R4)           INCREMENT COUNTER
         CH    R4,PARMLEN          IS COUNT = PARMLEN(W/O '/')
         BE    RESETPRM            YES, GO FIX PARM FIELD
         BCT   R6,PGMLOOP          LOOP TO CHECK AGAIN
         WTO   'CCD004I PROGRAM NAME SPECIFIED IN PARM FIELD IS LONGER X
               THAN 8 CHARACTERS-ABENDING',ROUTCDE=11
         ABEND 0004,DUMP,,USER     ABEND FOR BAD PROGRAM NAME
RESETPRM LR    R5,R4               SAVE LENGTH FOR LATER
         BCTR  R4,R0               DECREMENT FOR MVC EXECUTE
         EX    R4,MVCPGM           EXECUTE MOVE OF PROGRAM NAME
         LR    R4,R5               RESTORE LENGTH
         SR    R5,R5               CLEAR REGISTER
         LH    R5,PARMLEN          LOAD PARM LENGTH
         CR    R4,R5               IS COUNT= PARMLEN(WITHOUT ANY '/')
         BE    MOVEPARM            YES, SKIP INCREMENT
         LA    R4,1(,R4)           ADD 1 FOR ABOVE DECREMENT
```

```
MOVEPARM SR    R5,R5              CLEAR REGISTER
         LH    R5,PARMLEN         LOAD LENGTH OF ORIG PARMLIST
         SR    R5,R4              SUBTRACT LENGTH OF PGMNAME & '/'
         STH   R5,PARMLEN         STORE NEW PARM LENGTH
         LTR   R5,R5              WAS ANY OTHER PARM GIVEN
         BZ    ARNDSETF           NO, GO LINK TO PROGRAM
         CH    R5,=H'90'          IS REAL PARM GREATER THAN I CAN TAKE
         BNH   SETPARM            NO, I CAN HANDLE IT
         WTO   'CCD005I PARM FIELD FOR PROGRAM BEING INVOKED EXCEEDS ALX
               LOWABLE MAXIMUM OF 90 CHARACTERS-ABENDING',ROUTCDE=11
         ABEND 0005,DUMP,,USER    ABEND FOR BAD PROGRAM NAME
SETPARM  LA    R8,0(R4,R8)        LOAD A(WHATS AFTER 'PGMNAME/')
         LH    R4,PARMLEN         LOAD W/LENGTH FOR EXECUTE
         BCTR  R4,R0              DECREMENT FOR MVC EXECUTE
         EX    R4,NEWPARM         EXECUTE MOVE OF NEW PARM FIELD
ARNDSETF L     R14,CVTPTR         LOAD A(CVT)
         USING CVT,R14            ESTABLISH ADDRESSABILITY
         L     R14,CVTTCBP        LOAD A(CVTWORDS)
         L     R14,4(R14)         LOAD A(CURR TCB)
         USING TCB,R14            ESTABLISH ADDRESSABILITY
         L     R14,TCBTIO         LOAD A(TIOT)
         USING TIOT1,R14          ESTABLISH ADDRESSABILITY
         LA    R14,TIOENTRY       LOAD A(TIOT DD ENTRY)
         SR    R15,R15            CLEAR REGISTER
TIOTLOOP AR    R14,R15            BUMP POINTER
         IC    R15,0(R14)         INSERT DD ENTRY LENGTH
         LTR   R15,R15            LENGTH = 0
         BZ    TIOTDONE           YES, NO DD-ASSUME CC=00
         CLC   4(8,R14),=CL8'CONDCODE' IS IT DDNAME CONDCODE
         BNE   TIOTLOOP           NO, GET NEXT TIOT ENTRY
         OPEN  (CONDDCB,INPUT)    ELSE, OPEN CONTROL CARD FILE
         TM    CONDDCB+DCBOFLGS-IHADCB,DCBOFOPN WAS OPEN GOOD
         BO    GETREC             YES, GO READ CONTROL CARD
         WTO   'CCD008I OPEN FAILED FOR CONDCODE DD STATEMENT-ABENDING X
                ',ROUTCDE=11      ELSE, ISSUE ERROR MESSAGE
         ABEND 0008,DUMP,,USER    ABEND
GETREC   GET   CONDDCB,RECAREA    READ CONTROL CARD
         CLC   CCLIT,RECAREA      IS IT A VALID KEYWORD
         BE    GETCC              YES, GO CHECK VALUE
         WTO   'CCD010I INVALID CONTROL CARD FOUND-ABENDING',          X
               ROUTCDE=11         ELSE ISSUE ERROR MESSAGE
         ABEND 0010,DUMP,,USER    ABEND
GETCC    TRT   RECAREA+L'CCLIT(4),TRANTAB ARE THE 4 BYTES NUMERIC
         BZ    CODENUM            YES, GO USE THEM
         WTO   'CCD011I INVALID VALUE FOUND IN CONTROL CARD-ABENDING', X
               ROUTCDE=11         ELSE, ISSUE ERROR MESSAGE
         ABEND 0011,DUMP,,USER    ABEND
ENDCARD  WTO   'CCD012I NO VALID CONDCODE CONTROL CARD FOUND-ABENDING',X
               ROUTCDE=11         ISSUE ERROR MESSAGE
         ABEND 0012,DUMP,,USER    ABEND
CODENUM  CLOSE (CONDDCB,DISP)     CLOSE CONTROL CARD FILE
```

```
         PACK  PACKFLD,RECAREA+L'CCLIT(4) PACK COND CODE VALUE
         CVB   R1,PACKFLD         CONVERT TO BINARY
         ST    R1,CONDCHEK        STORE FOR LATER USE
* CHECK PGMTABLE FOR SPECIAL PGMS TO USE LINK INSTEAD OF ATTACH
TIOTDONE LA    R1,PGMTABLE        LOAD A(EXCLUSION TABLE)
CHKPGMTB CLC   0(8,R1),PGMTABND   ARE WE AT END OF TABLE
         BE    DOATTACH           YES, GO DO ATTACH
         CLC   0(8,R1),PGMNAME    ELSE, IS THIS AN EXCLUDED PGM
         BE    DOLINK             YES, GO DO LINK
         LA    R1,8(,R1)          ELSE, BUMP TO NEXT TABLE ENTRY
         B     CHKPGMTB           GO CHECK NEXT ENTRY
DOATTACH LA    R1,PARMPTR         LOAD A(NEW PARM)
         ATTACH EPLOC=PGMNAME,ECB=ECB,JSTCB=YES
         LTR   R15,R15            WAS ATTACH SUCCESSFUL
         BZ    SAVETCB            YES, CONTINUE
         WTO   'CCD006I ATTACH OF PROGRAM FAILED-ABENDING',ROUTCDE=11
         ABEND 0006,DUMP,,USER    ELSE, ABEND
DOLINK   LA    R1,PARMPTR         LOAD A(NEW PARM)
         LINK  EPLOC=PGMNAME      LINK TO REQUESTED PROGRAM
         B     TRANSRC            GO PROCESS RETURN CODE
SAVETCB  ST    R1,TCBADDR         SAVE A(ATTACHED TCB)
         WAIT  ECB=ECB            WAIT FOR TASK TO FINISH
         DETACH TCBADDR           DETACH TCB
         L     R15,ECB            LOAD ECB DATA (ABEND/COND CODE)
         N     R15,=X'00FFF000'   TURN OFF ALL BUT SYS ABEND FIELDS
         LTR   R15,R15            WAS THERE ANY SYSTEM ABEND
         BZ    LTR15              NO, GO CHECK USER ABEND/COND CODE
         SRL   R15,12             ELSE, SHIFT TO LOW ORDER
         ST    R15,ABENDCD        STORE ABEND CODE FOR LATER
         MVC   MSG009TP,=C'ABEND ' INDICATE ABEND IN MESSAGE
         B     TRANSRC            GO MAKE ABEND CODE PRINTABLE
LTR15    L     R15,ECB            LOAD ECB DATA (USER ABEND/COND CODE)
         N     R15,=X'00000FFF'   TURN OFF ALL BUT USER/COND CODE FIELD
         LTR   R15,R15            WAS IT NORMAL COMPLETION
         BZ    RETURN             YES, GO RETURN NORMALLY
TRANSRC  ST    R15,PGMCOND        SAVE INVOKED PROGRAM COND CODE
         ST    R15,HEXIN          ELSE, STORE RETURN CODE
         UNPK  HEXOUT(9),HEXIN(5) UNPACK THE HEX VALUES
         TR    HEXOUT(9),HEXTAB-X'F0' TRANSLATE
         MVC   MSG009NM,PGMNAME   MOVE PROG NAME TO MESSAGE
         MVC   MSG009CC,HEXOUT+4  MOVE RETURN CODE TO MESSAGE
         WTO   MF=(E,MSG009)      ISSUE RETURN CODE MESSAGE
         CLC   ABENDCD,=X'00000000' WAS AN ABEND DETECTED EARLIER
         BNE   ABENDSYS           YES, GO ISSUE SYSTEM ABEND
         L     R15,PGMCOND        RELOAD ACTUAL RETURNED COND CODE
         C     R15,CONDCHEK       COMPARE TO CODED COND CODE VALUE
         BNH   RETURN             IF NOT LOWER, THEN ACCEPT IT
         ABEND 0009,DUMP,,USER    ELSE, ABEND FOR NON-ZERO COND CODE
ABENDSYS L     R1,ABENDCD         LOAD CALLED PROGRAMS ABEND CODE
         ABEND (1),,,SYSTEM       ABEND FOR CALLED PROGRAM ABENDING
RETURN   L     R13,SAVEAREA+4     RELOAD A(CALLERS S/A)
```

```
         RETURN (14,12),RC=(15)   RETURN TO OS
MVCPGM   MVC   PGMNAME(0),0(R8)   EXECUTED MVC OF PGM NAME FOR LINK
NEWPARM  MVC   PARMSTRT(0),0(R8)  EXECUTED MVC OF REST OF PARM FIELD
SAVEAREA DC    18F'0'             OS SAVEAREA
TCBADDR  DC    F'0'               A(ATTACHED TCB)
PACKFLD  DC    D'0'               AREA FOR PACKING
CONDCHEK DC    F'0'               DEFAULT COND CODE TO CHECK FOR
PGMCOND  DC    F'0'               AREA FOR CALLED PGM'S COND CODE
ABENDCD  DC    F'0'               AREA FOR CALLED PGM'S ABEND CODE
CMPF     DC    XL4'00000000'      TCBCMP
ECB      DC    XL4'00000000'      ECB FOR ATTACH
PARMPTR  DC    X'80',AL3(PARM)    POINTER TO PARM FIELD
PGMNAME  DC    CL8' '             AREA FOR PROGRAM NAME TO LINK
PARM     DS    0F                 BEGINNING OF RMF PARM FIELD
PARMLEN  DC    AL2(PARMEND-PARMSTRT) HALFWORD OF PARM LENGTH
PARMSTRT DC    CL90' '            ACTUAL PARM FIELD
PARMEND  EQU   *                  EQUATE FOR END OF PARM FIELD
PGMTABLE DS    0F                 SPECIAL PROGRAM NAME TABLE ------|
         DC    CL8'ACPMAIN'       BMC UTILITY-COPY PLUS            |
         DC    CL8'ADUUMAIN'      BMC UTILITY-UNLOAD PLUS          |
         DC    CL8'AEXEMAIN'      BMC UTILITY-BMCTRIG              |
         DC    CL8'AFRMAIN'       BMC UTILITY-RECOVER PLUS         |
         DC    CL8'AMUUMAIN'      BMC UTILITY-LOAD PLUS            |
         DC    CL8'ARUUMAIN'      BMC UTILITY-REORG PLUS           |
         DC    CL8'ASUUMAIN'      BMC UTILITY-BMCSTATS             |
PGMTABND DC    X'FFFFFFFFFFFFFFFF'  END OF PROGRAM NAME TABLE -----|
TRANTAB  DC    256X'FF'           TRANSLATE TABLE
         ORG   TRANTAB+240        POINT BACK TO NUMERICS
         DC    10X'00'            NUMBERS 0-9
         ORG   ,                  RESET LOCATION COUNTER
HEXTAB   DC    X'F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6'  123456789ABCDEF
HEXIN    DS    F                  FOR HEXCONV ROUTINE.
         DS    CL1                REQUIRED AFTER HEXIN FOR BIT FLIP
HEXOUT   DS    CL8                FOR HEXCONV ROUTINE.
         DS    CL1                REQUIRED AFTER HEXOUT FOR BIT FLIP
MSG009   WTO   'CCD009I ******** SET RETURN CODE OF X''****''',       X
               ROUTCDE=(11),MF=L  RETURN CODE MESSAGE SKELETON
MSG009NM EQU   MSG009+12,8        OFFSET TO CALLED PROGRAM NAME
MSG009TP EQU   MSG009+25,6        OFFSET TO CALLED PROGRAM NAME
MSG009CC EQU   MSG009+42,4        OFFSET TO ABEND/COND CODE
RECAREA  DC    CL80' '            INPUT AREA FOR CONTROL CARD
CCLIT    DC    C'MAX-ALLOWABLE-COND-CODE=' CONTROL CARD KEYWORD
CONDDCB  DCB   RECFM=FB,DSORG=PS,LRECL=80,MACRF=GM,EODAD=ENDCARD,     X
               DDNAME=CONDCODE
         CVT   DSECT=YES,LIST=NO
         IKJTCB DSECT=YES,LIST=NO
TIOT     DSECT
         IEFTIOT1                 TIOT MAPPING
         DCBD  DSORG=PS
         END
```

Scheduling jobs and system commands

INTRODUCTION

Every MVS environment has a need to do time-of-day scheduling for batch job submission or operator command execution. Not all of these events require the sophistication of commercially available scheduling products. In a number of situations it's merely a case of,"if it's this time of day, then we need to do that" scenario. The scheduler described by this article provides for basic time-of-day scheduling for batch jobs or system commands.

The scheduler is composed of two programs – a driver program SCHEDM01, and a subtask program SCHEDM02. Both programs have authorization requirements so they should reside in an APF authorized library and be linked AC(1).

The SCHEDM01 program is responsible for interpreting the parameter library, for building the schedule, for establishing operator communication, and for attaching the SCHEDM02 program.

The SCHEDM02 program is responsible for checking the schedule that was built by SCHEDM01 and determining when the next scheduled event is to occur. The basic approach is to:

1 Determine what day of the week the program is operating in.

2 Check to see if any events are waiting to occur for that day.

3 If they are, do an STIMER wait for the time of day of the first event scheduled to occur that day. If there aren't any events waiting to occur for that day, do an STIMER wait for 23:59:59.

4 When the STIMER triggers, perform the required function and return to step 1.

The following sample JCL can be used when creating your scheduler:

```
//SCHEDULE EXEC PGM=SCHEDMØ1,TIME=144Ø
//SYSIN    DD DSN=SCHEDULE.PARMLIB,DISP=SHR
//JOBFILE  DD DSN=SCHEDULE.JOBFILE,DISP=SHR
//CMDFILE  DD DSN=SCHEDULE.CMDFILE,DISP=SHR
```

```
//CHKPOINT DD DSN=SCHEDULE.CHKPOINT,DISP=SHR
//INTRDR   DD SYSOUT=(A,INTRDR)
//SYSABEND DD SYSOUT=A
```

The datasets that are referenced above are used in the following ways:

- SYSIN – contains the parameter information used for scheduler start-up. This dataset has DCB characteristics of DSORG=PS, LRECL=80, RECFM=FB.
- JOBFILE – contains one member for each JOB referenced in SYSIN. Each JOB referenced in the JOB=*jobname* parameter must have a corresponding *jobname* member in the JOBFILE dataset. This dataset has DCB characteristics of DSORG=PO, LRECL=80, RECFM=FB.
- CMDFILE – contains one member for each CMD referenced in SYSIN. Each CMD referenced in the CMD=*cmdname* parameter must have a corresponding *cmdname* member in the CMDFILE dataset. This dataset has DCB chracteristics of DSORG=PO, LRECL=80, RECFM=FB.
- CHKPOINT – contains the scheduler checkpoint record. This dataset has DCB characteristics mDSORG=PS, LRECL=80, RECFM=F. For initial scheduler start-up, this file should contain a single record, in columns 1 through 16, of the following format:

  ```
  CHKPOINTCHKPOINT
  ```

  This indicates to the scheduler that this is a first-time start-up and that RECOVER processing should not take place.

Here is some example parameter data that could be included in the SYSIN dataset:

```
* ALL COMMENT CARDS START WITH A ASTERISK
* NOTE: FOR JOBMAX AND TIME VALUES, ALL FOUR DIGITS MUST BE CODED
*
* JOBMAX SPECIFIES THE MAXIMUM NUMBER OF JOB= KEYWORDS THAT FOLLOW
*
* CHECKPOINTINTERVAL SPECIFIES THE NUMBER OF MINUTES BETWEEN
*    CHECKPOINT UPDATES (60 IS THE MAXIMUM)
JOBMAX=0200,CHECKPOINTINTERVAL=10
*
* COLUMN  1 MUST CONTAIN THE DAY OF THE WEEK OR 'EVERYDAY'
* COLUMN 10 MUST CONTAIN 'T=' FOLLOWED BY THE FOUR DIGIT TIME VALUE
* COLUMN 16 CAN CONTAIN 'JOB=' FOLLOWED BY THE JOBFILE MEMBER NAME OR
```

```
* COLUMN 16 CAN CONTAIN 'CMD=' FOLLOWED BY THE CMDFILE MEMBER NAME
* COLUMN 29 CAN CONTAIN THE 'CONTINUE=' KEYWORD FOLLOWED BY A FOUR
*           DIGIT TIME VALUE (HHMM)
*           THIS PARAMETER REPRESENTS THE TIME INTERVAL FROM THE 'T='
*           TIME AT WHICH THE JOB OR COMMAND IS REISSUED
* COLUMN 43 CAN CONTAIN THE 'RECOVER' KEYWORD
*           THIS PARAMETER INDICATES WHETHER THE JOB OR COMMAND IS
*           TO BE ISSUED IF THE TIME IT WAS SUPPOSED TO OCCUR IS
*           WITHIN THE LAST 24 HRS AND SCHEDULER WAS DOWN AT THAT TIME
*
* ISSUE THE COMMANDS IN THE RMFSTOP MEMBER OF CMDFILE AT Ø4:Ø5 A.M.
* EVERYDAY.
EVERYDAY  T=Ø4Ø5,CMD=RMFSTOP
*
* ISSUE THE COMMANDS IN THE CMDSØ53Ø MEMBER OF CMDFILE AT Ø5:3Ø AM.
* EVERY WEEKDAY DAY AND PROCESS THESE COMMANDS IF SCHEDULER FAILED
* PRIOR TO Ø5:3Ø AND WAS STARTED AFTER THAT TIME (RECOVER).
MONDAY    T=Ø53Ø,CMD=CMDSØ53Ø               RECOVER
TUESDAY   T=Ø53Ø,CMD=CMDSØ53Ø               RECOVER
WEDNESDAY T=Ø53Ø,CMD=CMDSØ53Ø               RECOVER
THURSDAY  T=Ø53Ø,CMD=CMDSØ53Ø               RECOVER
FRIDAY    T=Ø53Ø,CMD=CMDSØ53Ø               RECOVER
*
* SUBMIT THE JOB IN THE WRITELOG MEMBER OF JOBFILE AT ØØ:ØØ EVERYDAY
* AND RE-SUBMIT THAT SAME JOB EVERY 9Ø MINUTES (CONTINUE=Ø13Ø)
* THROUGHOUT THE DAY.
EVERYDAY  T=ØØØØ,JOB=WRITELOG CONTINUE=Ø13Ø
*
```

OPERATIONAL ENVIRONMENT

SCHEDM01 and SCHEDM02 have been tested and run in both an MVS/ESA 4.3 environment and an OS/390 Version 1 Release 2 environment.

SCHEDM01

```
*   ###  SUMMARY OF CHANGES
*
*   CHANGE IDENTIFIER       CHANGE DESCRIPTION
*
*   C9Ø197                  ADDED FREE TO LAST CLOSE OF SYSIN
*   C9Ø2Ø1                  ADDED SUPPORT FOR CHECKPOINTING DATASET
*                           AND RECOVER KEYWORD FOR CATCHUP PROCESSING
*   C941Ø8                  ADDED SUPPORT FOR MODIFY 'STOP' COMMAND
*
```

```
*************************************************************************
*                                                                       *
*   THE SCHEDMØ1 TASK READS THE SYSIN DATASET LOOKING FOR THE           *
*   JOBMAX/CHECKPOINTINTERVAL CONTROL CARD.  THE JOBMAX PARAMETER       *
*   DEFINES THE MAXIMUM NUMBER OF JOB=/CMD= KEYWORDS THAT CAN EXIST     *
*   IN THE PARMLIB DATASET.  FOR EXAMPLE:                               *
*                                                                       *
*   JOBMAX=Ø2ØØ,CHECKPOINTINTERVAL=1Ø                                   *
*                                                                       *
*   KEEP IN MIND THAT KEYWORD OPTIONS SUCH AS 'EVERYDAY' OR 'CONTINUE'* *
*   AFFECT THE JOBMAX TOTAL.  FOR EXAMPLE, IF YOU CODE:                 *
*                                                                       *
*   EVERYDAY  T=ØØØØ,JOB=JOB1     CONTINUE=Ø2ØØ                         *
*                                                                       *
*   YOU WOULD REQUIRE A MINIMUM JOBMAX OF 84 (7*12 - SEVEN DAYS/WEEK    *
*   * 12 TIMES/DAY (EVERY TWO HOURS AS PER THE CONTINUE=Ø2ØØ PARM).     *
*                                                                       *
*   THE CHECKPOINTINTERVAL PARAMETER DEFINES HOW OFTEN THE SCHEDMØ1     *
*   MODULE SHOULD UPDATE THE CHECKPOINT.  THE DEFAULT IS EVERY          *
*   FIVE MINUTES.  THE MAXIMUM IS EVERY 6Ø MINUTES.  THE CHECKPOINT     *
*   IS USED TO DETERMINE IF THE SCHEDULER WAS TERMINATED AS THE         *
*   RESULT OF A SYSTEM OUTAGE.                                          *
*                                                                       *
*   IF YOU ARE STARTING SCHEDULER FOR THE FIRST TIME, THE CHECKPOINT    *
*   DATASET SHOULD BE INITIALIZED WITH AN 8Ø BYTE RECORD THAT CONTAINS* *
*   THE FOLLOWING CHARACTER SEQUENCE:                                   *
*                                                                       *
*   CHKPOINTCHKPOINT                                                    *
*                                                                       *
*   IN COLUMNS 1-16.  THIS INDICATES TO THE SCHEDULER THAT THIS IS A    *
*   FIRST TIME START-UP AND THAT NO ATTEMPT WILL BE MADE TO DO          *
*   'RECOVER' PROCESSING.  THIS CHECKPOINT RECORD WILL BE USED ON       *
*   FUTURE START-UP SITUATIONS TO DETERMINE IF THE SCHEDULER            *
*   TERMINATED NORMALLY OR NOT, AND IF ANY RECOVERY PROCESSING SHOULD   *
*   OCCUR.  THE FIRST EIGHT BYTES ARE USED FOR A HEARTBEAT TIMESTAMP    *
*   AND THE SECOND EIGHT BYTES ARE USED FOR A TERMINATION TIMESTAMP.    *
*   IF THE HEARTBEAT TIMESTAMP IS GREATER THAN THE TERMINATION          *
*   TIMESTAMP, THIS INDICATES A NON-NORMAL SHUTDOWN.                    *
*                                                                       *
*   SCHEDMØ1 NEEDS TO RESIDE IN AN APF AUTHORIZED LIBRARY AND BE        *
*   LINKED AC(1)                                                        *
*                                                                       *
*************************************************************************
         PRINT ON,GEN
         SPACE
SCHEDMØ1 CSECT
         $SETR
         USING SCHEDMØ1,R12,R11
         STM   R14,R12,12(R13)
         LR    R12,R15                 SET INITIAL BASE
         LA    R11,4Ø95(,R12)
```

```
         LA    R11,1(,R11)
         ST    R13,SAVE+4
         LA    R13,SAVE
*************************************************************************
SKIPID   OPEN  SYSIN                   OPEN SYSIN DATASET
         LA    R1Ø,SYSIN               ADDRESS DCB
         USING IHADCB,R1Ø
         TM    DCBOFLGS,X'1Ø'          OPEN OK?
         BZ    OPENERR1                NO - INDICATE ERROR
         DROP  R1Ø
*
SYSINLP1 GET   SYSIN                   READ A RECORD
         CLC   Ø(7,R1),=C'JOBMAX='     JOB MAX CARD?
         BE    JOBMAXFN                YES - PROCESS
         B     SYSINLP1                NO - TRY NEXT
*
EOD1     CLOSE (SYSIN)                 CLOSE SYSIN DATASET
         L     R3,=F'1ØØ'              SET DEFAULT JOB MAX
         L     R8,=F'5'                SET DEFAULT CHKPOINT INT.
         B     STRGCAL                 AND CONTINUE
*
JOBMAXFN PACK  DOUBLE+5(3),7(4,R1)     YES - THEN PACK MAX JOB NUMBER
         CVB   R3,DOUBLE               CONVERT TO BINARY
         XC    DOUBLE(8),DOUBLE        CLEAR THE AREA
         L     R8,=F'5'                SET DEFAULT CHKPOINT INT.
         CLC   12(19,R1),=C'CHECKPOINTINTERVAL=' LONG KEYWORD?
         BNE   STRGCAL                 NO - PROCESS DEFAULT
         TM    31(R1),X'FØ'            NUMERIC?
         BNO   CHKPTERR                NO - ISSUE ERROR WTO
         TM    32(R1),X'FØ'            NUMERIC?
         BNO   CHKPTERR                NO - ISSUE ERROR WTO
         PACK  DOUBLE+6(2),31(2,R1)    PACK CHECKPOINT INTERVAL
         CVB   R8,DOUBLE               CONVERT TO BINARY
         C     R8,=F'6Ø'               MORE THAN AN HOUR?
         BNH   STRGCAL                 NO - EVERYTHING IS OK
         WTO   'SCHEDØ1Ø - CHECKPOINT INTERVAL REQUESTED IS MORE THAN 6X
               Ø MINUTES.  MAXIMUM IS 6Ø MINUTES.  RESET TO MAXIMUM.', X
               ROUTCDE=(1),DESC=(1)
         L     R8,=F'6Ø'               SET MAX. CHKPOINT INT.
         B     STRGCAL                 GO GET STORAGE
CHKPTERR EQU   *
         WTO   'SCHEDØ11 - CHECKPOINT INTERVAL REQUESTED IS NOT A NUMERX
               IC VALUE.  A 5 MINUTE DEFAULT IS USED.',               X
               ROUTCDE=(1),DESC=(1)
         L     R8,=F'5'                SET DEFAULT CHKPOINT INT.
STRGCAL  LR    R2,R3                   SAVE AMOUNT
         MH    R3,=H'38'               GET AMOUNT OF STORAGE REQUIRED
         LR    R9,R2                   SAVE JOBMAX
         ST    R8,CHKPTINT             SAVE CHECKPOINT INTERVAL
         GETMAIN RC,LV=(R3)            GET STORAGE
         LTR   R15,R15                 OK?
```

```
         BNZ   GETERR                  NO - BIG PROBLEM
         LR    R7,R1                   ADDRESS STORAGE
         AR    R7,R3                   ADDRESS LAST BYTE
         ST    R7,STORLIM              SAVE LAST BYTE ADDRESS
         LR    R4,R1                   ADDRESS STORAGE
         LR    R5,R3                   GET LENGTH
         SR    R6,R6                   DUMMY ADDRESS
         SR    R7,R7                   ZERO SECOND LENGTH
         MVCL  R4,R6                   CLEAR AREA
         MH    R2,=H'4'                GET POINTER STORAGE SIZE
         ST    R1,DAYPTRS+Ø            STORE SUNDAY POINTER
         AR    R1,R2                   ADDRESS NEXT DAY
         ST    R1,DAYPTRS+4            STORE MONDAY POINTER
         AR    R1,R2                   ADDRESS NEXT DAY
         ST    R1,DAYPTRS+8            STORE TUESDAY POINTER
         AR    R1,R2                   ADDRESS NEXT DAY
         ST    R1,DAYPTRS+12           STORE WEDNESDAY POINTER
         AR    R1,R2                   ADDRESS NEXT DAY
         ST    R1,DAYPTRS+16           STORE THURSDAY POINTER
         AR    R1,R2                   ADDRESS NEXT DAY
         ST    R1,DAYPTRS+2Ø           STORE FRIDAY POINTER
         AR    R1,R2                   ADDRESS NEXT DAY
         ST    R1,DAYPTRS+24           STORE SATURDAY POINTER
         MVC   NXTDPTRS(28),DAYPTRS    NEXT AVAIL IS SET TO FIRST
         AR    R1,R2                   ADDRESS FREE AREA
         ST    R1,JOBPTRS              STORE JOB AREA ADDRESS
         ST    R1,NXTJPTRS             NEXT AVAIL IS SET TO FIRST
         LR    R1,R9                   GET JOBMAX VALUE
         SLL   R1,2                    MULTIPLY BY FOUR
         ST    R1,DAYLEN               SAVE LENGTH OF DAY AREA STORAGE
*
         A     R9,=F'1'                ADD ONE TO JOBMAX
         SLL   R9,4                    MULTIPLY BY SIXTEEN
         GETMAIN RC,LV=(R9)            GET STORAGE
         LTR   R15,R15                 GOT IT OK?
         BNZ   GETERR                  NO - WE BETTER QUIT
         ST    R1,RECOVTBL             SAVE TABLE ADDRESS
         LR    RØ,R1                   MOVE ADDRESS
         LR    R1,R9                   SET LENGTH
         XR    R14,R14                 DUMMY ADDRESS
         XR    R15,R15                 SET PATTERN BYTE
         MVCL  RØ,R14                  CLEAR STORAGE AREA
*
         OPEN  (CHKPOINT,INPUT)        OPEN CHECKPOINT DATASET
         TM    CHKPOINT+48,X'1Ø'       OPEN OK?
         BZ    OPENERR3                NO - ISSUE ERROR MESSAGE
         GET   CHKPOINT,CHKPTREC       READ CHECKPOINT RECORD
         CLC   CHKPTREC(8),=C'CHKPOINT' CHKPOINT INIT RECORD?
         BE    CHKPTOFF                YES - THE FIRST TIME
         CLC   CHKPTREC+8(8),=C'CHKPOINT' CHKPOINT INIT RECORD?
```

```
         BE    CHKPTOFF                YES - THE FIRST TIME
         OI    FLAG,ACTIVE             SET ACTIVE FLAG
CHKPTOFF CLOSE CHKPOINT                CLOSE CHECKPOINT DATASET
*************************************************************************
         MVC   SYSIN+33(3),=AL3(EOD2)  SET NEW EOD ADDRESS
         OPEN  SYSIN                   OPEN SYSIN DATASET
         LA    R1Ø,SYSIN               ADDRESS DCB
         USING IHADCB,R1Ø
         TM    DCBOFLGS,X'1Ø'          OPEN OK?
         BZ    OPENERR1                NO - INDCIATE ERROR
         DROP  R1Ø
SYSINLP2 GET   SYSIN                   READ A RECORD
         LR    R1Ø,R1                  ADDRESS INPUT RECORD
         USING PARMIN,R1Ø              ADDRESSIBILITY
         CLI   COMMENT,C'*'            COMMENT?
         BE    SYSINLP2                YES - TRY NEXT
         NI    FLAG,255-DAILY          CLEAR FLAG
         CLC   TIMEID,=C'T='           A VALID TIME PARM?
         BNE   BADPARM                 NO - INDICATE SO
         TM    RELTIME,X'FØ'           NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         TM    RELTIME+1,X'FØ'         NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         TM    RELTIME+2,X'FØ'         NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         TM    RELTIME+3,X'FØ'         NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         PACK  RELVALUE,RELTIME        PACK RELEASE TIME
         CLC   JOBID,=C'JOB='          JOBFILE MEMBER?
         BE    CHKFIELD                YES - GO CHECK FIELD INFO
         CLC   JOBID,=C'CMD='          CMDFILE MEMBER?
         BNE   BADPARM                 NO - ISSUE ERROR
CHKFIELD EQU   *
         CLI   JOB,C' '                JOB PRESENT?
         BE    BADPARM                 NO - BAD STUFF
         NI    FLAG,255-CON            TURN FLAG OFF
         CLC   CONTID,=C'CONTINUE='    CONTINUE REQUEST?
         BNE   NOCONT                  NO - JUST SKIP IT
         TM    CONTIME,X'FØ'           NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         TM    CONTIME+1,X'FØ'         NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         TM    CONTIME+2,X'FØ'         NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         TM    CONTIME+3,X'FØ'         NUMERIC?
         BNO   BADPARM                 NO - BAD STUFF
         PACK  CONVALH,CONTIME(2)      PACK CONTINUE TIME HOURS
         MP    CONVALH,=P'1ØØ'         PUSH HOURS OVER
         PACK  CONVALM,CONTIME+2(2)    PACK CONTINUE TIME MINUTES
         OI    FLAG,CON                YES - SET FLAG
```

```
NOCONT   L     R1,NXTJPTRS             ADDRESS NEXT AVAILABLE SPOT
         C     R1,STORLIM              ALL STORAGE BEEN USED?
         BNL   STORUSE                 YES - THEN NO MORE
         CLC   DAY,=CL9'EVERYDAY '     DAILY JOB?
         BNE   CHK2                    NO - GO CHECK ON NEXT
         OI    FLAG,DAILY              YES - SET FLAG
         B     SUNSET                  AND GO SET POINTERS
CHK2     CLC   DAY,=CL9'SUNDAY   '     SUNDAY JOB?
         BE    SUNSET                  YES - GO SET UP
         CLC   DAY,=CL9'MONDAY   '     MONDAY JOB?
         BE    MONSET                  YES - GO SET UP
         CLC   DAY,=CL9'TUESDAY  '     TUESDAY JOB?
         BE    TUESET                  YES - GO SET UP
         CLC   DAY,=CL9'WEDNESDAY'     WEDNESDAY JOB?
         BE    WEDSET                  YES - GO SET UP
         CLC   DAY,=CL9'THURSDAY '     THURSDAY JOB?
         BE    THUSET                  YES - GO SET UP
         CLC   DAY,=CL9'FRIDAY   '     FRIDAY JOB?
         BE    FRISET                  YES - GO SET UP
         CLC   DAY,=CL9'SATURDAY '     SATURDAY JOB?
         BE    SATSET                  YES - GO SET UP
         B     BADPARM
SUNSET   L     R2,NXTDPTRS+Ø           GET NEXT FREE POINTER AREA
         ST    R1,Ø(,R2)               SET POINTER
*
         CLC   RECOVID(7),=C'RECOVER'  RECOVER ON?
         BNE   NORECOV1                NO - DON'T SET RECOVER
         OI    Ø(R2),X'8Ø'             SET FLAG IN POINTER
NORECOV1 EQU   *
         CLC   JOBID,=C'CMD='          A COMMAND?
         BNE   NOCMD1                  NO - DON'T SET COMMAND
         OI    Ø(R2),X'4Ø'             SET FLAG IN POINTER
NOCMD1   EQU   *
*
         LA    R2,4(,R2)               ADDRESS NEXT FREE
         ST    R2,NXTDPTRS+Ø           AND SAVE NEXT FREE ADDRESS
         TM    FLAG,DAILY              DAILY JOB?
         BNO   JOBIN                   NO - THEN GO BRING JOB NAME IN
*
MONSET   L     R2,NXTDPTRS+4           GET NEXT FREE POINTER AREA
         ST    R1,Ø(,R2)               SET POINTER
*
         CLC   RECOVID(7),=C'RECOVER'  RECOVER ON?
         BNE   NORECOV2                NO - DON'T SET RECOVER
         OI    Ø(R2),X'8Ø'             SET FLAG IN POINTER
NORECOV2 EQU   *
         CLC   JOBID,=C'CMD='          A COMMAND?
         BNE   NOCMD2                  NO - DON'T SET COMMAND
         OI    Ø(R2),X'4Ø'             SET FLAG IN POINTER
NOCMD2   EQU   *
```

```
*
         LA    R2,4(,R2)               ADDRESS NEXT FREE
         ST    R2,NXTDPTRS+4           AND SAVE NEXT FREE ADDRESS
         TM    FLAG,DAILY              DAILY JOB?
         BNO   JOBIN                   NO - THEN GO BRING JOB NAME IN
TUESET   L     R2,NXTDPTRS+8           GET NEXT FREE POINTER AREA
         ST    R1,Ø(,R2)               SET POINTER
*
         CLC   RECOVID(7),=C'RECOVER'  RECOVER ON?
         BNE   NORECOV3                NO - DON'T SET RECOVER
         OI    Ø(R2),X'8Ø'             SET FLAG IN POINTER
NORECOV3 EQU   *
         CLC   JOBID,=C'CMD='          A COMMAND?
         BNE   NOCMD3                  NO - DON'T SET COMMAND
         OI    Ø(R2),X'4Ø'             SET FLAG IN POINTER
NOCMD3   EQU   *
*
         LA    R2,4(,R2)               ADDRESS NEXT FREE
         ST    R2,NXTDPTRS+8           AND SAVE NEXT FREE ADDRESS
         TM    FLAG,DAILY              DAILY JOB?
         BNO   JOBIN                   NO - THEN GO BRING JOB NAME IN
WEDSET   L     R2,NXTDPTRS+12          GET NEXT FREE POINTER AREA
         ST    R1,Ø(,R2)               SET POINTER
*
         CLC   RECOVID(7),=C'RECOVER'  RECOVER ON?
         BNE   NORECOV4                NO - DON'T SET RECOVER
         OI    Ø(R2),X'8Ø'             SET FLAG IN POINTER
NORECOV4 EQU   *
         CLC   JOBID,=C'CMD='          A COMMAND?
         BNE   NOCMD4                  NO - DON'T SET COMMAND
         OI    Ø(R2),X'4Ø'             SET FLAG IN POINTER
NOCMD4   EQU   *
*
         LA    R2,4(,R2)               ADDRESS NEXT FREE
         ST    R2,NXTDPTRS+12          AND SAVE NEXT FREE ADDRESS
         TM    FLAG,DAILY              DAILY JOB?
         BNO   JOBIN                   NO - THEN GO BRING JOB NAME IN
THUSET   L     R2,NXTDPTRS+16          GET NEXT FREE POINTER AREA
         ST    R1,Ø(,R2)               SET POINTER
*
         CLC   RECOVID(7),=C'RECOVER'  RECOVER ON?
         BNE   NORECOV5                NO - DON'T SET RECOVER
         OI    Ø(R2),X'8Ø'             SET FLAG IN POINTER
NORECOV5 EQU   *
         CLC   JOBID,=C'CMD='          A COMMAND?
         BNE   NOCMD5                  NO - DON'T SET COMMAND
         OI    Ø(R2),X'4Ø'             SET FLAG IN POINTER
NOCMD5   EQU   *
*
         LA    R2,4(,R2)               ADDRESS NEXT FREE
```

```
         ST    R2,NXTDPTRS+16          AND SAVE NEXT FREE ADDRESS
         TM    FLAG,DAILY              DAILY JOB?
         BNO   JOBIN                   NO - THEN GO BRING JOB NAME IN
*
FRISET   L     R2,NXTDPTRS+2Ø          GET NEXT FREE POINTER AREA
         ST    R1,Ø(,R2)               SET POINTER
*
         CLC   RECOVID(7),=C'RECOVER'  RECOVER ON?
         BNE   NORECOV6                NO - DON'T SET RECOVER
         OI    Ø(R2),X'8Ø'             SET FLAG IN POINTER
NORECOV6 EQU   *
         CLC   JOBID,=C'CMD='          A COMMAND?
         BNE   NOCMD6                  NO - DON'T SET COMMAND
         OI    Ø(R2),X'4Ø'             SET FLAG IN POINTER
NOCMD6   EQU   *
*
         LA    R2,4(,R2)               ADDRESS NEXT FREE
         ST    R2,NXTDPTRS+2Ø          AND SAVE NEXT FREE ADDRESS
         TM    FLAG,DAILY              DAILY JOB?
         BNO   JOBIN                   NO - THEN GO BRING JOB NAME IN
SATSET   L     R2,NXTDPTRS+24          GET NEXT FREE POINTER AREA
         ST    R1,Ø(,R2)               SET POINTER
*
         CLC   RECOVID(7),=C'RECOVER'  RECOVER ON?
         BNE   NORECOV7                NO - DON'T SET RECOVER
         OI    Ø(R2),X'8Ø'             SET FLAG IN POINTER
NORECOV7 EQU   *
         CLC   JOBID,=C'CMD='          A COMMAND?
         BNE   NOCMD7                  NO - DON'T SET COMMAND
         OI    Ø(R2),X'4Ø'             SET FLAG IN POINTER
NOCMD7   EQU   *
*
         LA    R2,4(,R2)               ADDRESS NEXT FREE
         ST    R2,NXTDPTRS+24          AND SAVE NEXT FREE ADDRESS
JOBIN    SR    R9,R9                   CLEAR REG
         ICM   R9,B'Ø111',RELVALUE     GET RELEASE TIME
         SRL   R9,4                    DROP SIGN
         STH   R9,Ø(,R1)               STORE RELEASE TIME
         MVC   2(8,R1),JOB             BRING IN JOB NAME
         LA    R1,1Ø(,R1)              ADDRESS NEXT AVAILABLE
         ST    R1,NXTJPTRS             AND STORE IT
         TM    FLAG,CON                CONTINUE?
         BNO   SYSINLP2                NO - GET NEXT CARD
         SR    R9,R9                   CLEAR REGISTER
         ICM   R9,B'ØØ11',RELVALUE+1   GET HMMF FROM ØHHMMF
         N     R9,=X'ØØØØØFFF'         GET ØMMF FROM HMMF
         XC    WORK,WORK               CLEAR AREA
         STH   R9,WORK+3               STORE ØMMF
         AP    WORK,CONVALM            ADD CONTINUE MINUTES
         DP    WORK,=PL2'6Ø'           GET MINUTES AND HOURS
```

```
*   WORK -> 0000HF,0MMF
         NC    RELVALUE+1(2),=X'F00F'  CLEAR PREVIOUS MINUTES
         OC    RELVALUE+1(2),WORK+3    SET NEW MINUTES
         MP    WORK(3),=P'100'         SHIFT HOURS OVER
         AP    RELVALUE,WORK(3)        ADD ON HOUR FROM MINUTES ADD
         AP    RELVALUE,CONVALH        ADD ON CONTINUE TIME HOURS
         CP    RELVALUE,=P'2359'       PAST END OF THE DAY?
         BNH   NOCONT                  NO - THEN CONTINUE
         B     SYSINLP2                GET NEXT CARD
*****************************************************************
BADPARM  MVC   BADPMSG+34(30),DAY      MOVE PARM OUT
BADPMSG  WTO   'SCHED012 - PARM IGNORED - X                          X
                ',ROUTCDE=(1),DESC=(1)
         B     SYSINLP2                GET NEXT CARD
*****************************************************************
STORUSE  MVC   STORWTO+41(8),JOB       MOVE JOB NAME OUT
         CLC   JOBID,=C'CMD='          IS IT A COMMAND?
         BNE   STORWTO                 NO - GO ISSUE THE MSG AS IS
         MVC   STORWTO+37(3),=C'CMD'   MOVE JOB NAME OUT
STORWTO  WTO   'SCHED013 - JOBMAX EXCEEDED - JOB XXXXXXXX AND FOLLOWINGX
                JOBS/CMDS IGNORED',ROUTCDE=(1),DESC=(1)
EOD2     CLOSE (SYSIN,FREE)            CLOSE SYSIN DATASET
         DROP  R10
*****************************************************************
         L     R1,=F'0'                SET FOR FIRST DAY
SORTSTRT L     R6,NXTDPTRS(R1)         ADDRESS END OF LIST
         S     R6,=F'4'                LESS ONE ENTRY
         L     R5,DAYPTRS(R1)          ADDRESS START OF LIST
SORT     L     R3,0(,R5)               ADDRESS FIRST ENTRY
         L     R4,4(,R5)               ADDRESS NEXT ENTRY
         CLC   0(2,R3),0(R4)           NEXT ENTRY LESS THAN THIS
         BNH   NOSWAP                  NO - SKIP SWITCH
         ST    R3,4(,R5)               NEXT BECOMES FIRST
         ST    R4,0(,R5)               FIRST BECOMES NEXT
NOSWAP   A     R5,=F'4'                ADDRESS NEXT SORT ENTRY
         CR    R5,R6                   AT END OF LIST?
         BL    SORT                    NO - CONTINUE
         S     R6,=F'4'                LESS ONE ENTRY
         C     R6,DAYPTRS(R1)          FINISHED?
         BNH   FINISHED                YES - PUT ENTRIES BACK IN BLOCKS
         L     R5,DAYPTRS(R1)          ADDRESS START OF LIST
         B     SORT                    AND DO SOME MORE
FINISHED C     R1,=F'24'               COMPLETED ALL DAYS?
         BE    GETDAY                  YES - THEN FINISHED
         LA    R1,4(,R1)               ADDRESS NEXT DAY
         B     SORTSTRT                AND CONTINUE
*****************************************************************
*   DAY OF WEEK CALC
GETDAY   TIME  DEC                     GET TIME AND DATE
         ST    R1,DATE                 STORE DATE
```

```
         ST    RØ,TIME                  STORE TIME
*   INDEX YEAR TABLE
GETDAY2  LH    R2,DATE                  LOAD YEAR
         LA    R3,YEARTAB               ADDRESS YEAR TABLE
YEARSRCH CLM   R2,B'ØØØ1',Ø(R3)         YEAR FOUND?
         BE    YEARFND                  YES - GO PROCESS
         CLI   Ø(R3),X'FF'              NO - END OF TABLE?
         BE    EXPIRED                  YES - THE PROGRAM TABLE EXPIRED
         LA    R3,2(,R3)                NO - ADDRESS NEXT ENTRY
         B     YEARSRCH
YEARFND  SR    R2,R2                    CLEAR REGISTER
         IC    R2,1(,R3)                GET STARTING DAY OF YEAR
         XC    DOUBLE,DOUBLE            CLEAR DOUBLEWORD
         MVC   DOUBLE+6(2),DATE+2       MOVE IN DAY
         CVB   R1,DOUBLE                CONVERT DAY TO BINARY
         SR    RØ,RØ                    CLEAR EVEN REGISTER
         D     RØ,=F'7'                 DIVIDE BY 7 (DAYS IN A WEEK)
         LR    R1,RØ                    MOVE REMAINDER
         S     R1,=F'1'                 REMAINDER MINUS ONE
         AR    R1,R2                    PLUS STARTING DAY OF YEAR
         C     R1,=F'-1'                IS IT NEGATIVE?
         BNE   GETDAY3                  NO - THEN NO PROBLEMS
         L     R1,=F'6'                 SET TO SATURDAY
GETDAY3  IC    R1,DAYTABLE(R1)          GET CURRENT DAY OF THE WEEK
         STC   R1,CURRDAY               AND STORE IT
         MVC   JULDAY,DATE+2            SAVE JULIAN DAY
*****************************************************************
*   FIND FIRST JOB TO BE SUBMITTED
         SR    R3,R3                    CLEAR REGISTER
         IC    R3,CURRDAY               GET CURRENT DAY
         MH    R3,=H'4'                 GET DAY POINTER OFFSET
         L     R2,DAYPTRS(R3)           ADDRESS TODAYS WORK
*
         TM    FLAG,ACTIVE              CHKPOINT ACTIVE?
         BZ    SETTIMER                 NO - DON'T RECOVER
*
         MVC   TEMPDAT1(4),CHKPTREC+4   MOVE THE DATE
         MVC   TEMPDAT1+4(4),CHKPTREC   MOVE THE TIME
         MVC   TEMPDAT2(4),CHKPTREC+12  MOVE THE DATE
         MVC   TEMPDAT2+4(4),CHKPTREC+8 MOVE THE TIME
         CLC   TEMPDAT2(8),TEMPDAT1     TERM TIME>CHKPT TIME
         BH    TERMSOFT                 YES - SOFT SHUTDOWN
         OI    FLAG,HARDDOWN            SET HARD DOWN FLAG
TERMSOFT EQU   *
         MVC   CHKPTREC(8),CHKPTREC+8   MOVE IN TERMINATION TIME
*
         TIME  DEC                      GET CURRENT DATE & TIME
         STM   RØ,R1,CURRTIME           SAVE IT
         CLC   CURRDATE(4),CHKPTREC+4   SAME DATE?
         BNE   DAYWRAP                  NO - DAY WRAP SINCE CHKPT
```

```
         L     R8,DAYPTRS(R3)          ADDRESS CURRENT DAYS WORK
         L     R14,DAYLEN              GET MAX DAY STORAGE
         LA    R14,Ø(R14,R8)           SET TO END OF DAY
         ST    R14,DAYEND              SAVE IT
JOBLOOP1 L     R1Ø,Ø(,R8)              GET FIRST UNIT OF WORK
         LTR   R1Ø,R1Ø                 ANYTHING?
         BZ    RECOVEND                NO - NO MORE TO DO
         C     R8,DAYEND               DONE THE DAY?
         BE    RECOVEND                YES - NO MORE TO DO
         TM    Ø(R8),X'8Ø'             RECOVER FLAG?
         BZ    NEXTJOB1                NO - GET NEXT JOB
         CLC   Ø(2,R1Ø),CHKPTREC       BEFORE LAST CHECKPOINT?
         BNH   NEXTJOB1                YES - DON'T RECOVER
         CLC   Ø(2,R1Ø),CURRTIME       AFTER CURRENT TIME?
         BNL   RECOVEND                YES - THAT'S ALL WE NEED
         L     R9,RECOVTBL             GET RECOVERY TBL ADDR
TBLLOOP1 CLC   Ø(8,R9),=8X'ØØ'         BLANK ENTRY?
         BE    ADDJOB1                 YES - GO ADD TO RECOV TBL
         CLC   Ø(8,R9),2(R1Ø)          SAME JOB?
         BE    CHKFLAG1                YES - GO SEE IF CMD OR JOB
         LA    R9,16(,R9)              POINT TO NEXT ENTRY
         B     TBLLOOP1                GO CHECK IT OUT
ADDJOB1  MVC   Ø(8,R9),2(R1Ø)          MOVE IN JOB
         TM    Ø(R8),X'4Ø'             A COMMAND?
         BZ    SETJOB1                 NO - SET JOB FLAG
SETCMD1  OI    8(R9),X'4Ø'             SET FLAG IN RECOVER AREA
         B     NEXTJOB1                GO CHECK NEXT ENTRY
SETJOB1  OI    8(R9),X'8Ø'             SET FLAG IN RECOVER AREA
NEXTJOB1 LA    R8,4(,R8)               POINT TO NEXT ENTRY
         B     JOBLOOP1                CHECK IT OUT
CHKFLAG1 EQU   *
         TM    8(R9),X'CØ'             BOTH FLAGS SET ALREADY?
         BO    NEXTJOB1                YES - CAN'T DO ANY MORE
         TM    Ø(R8),X'4Ø'             A COMMAND?
         BO    SETCMD1                 YES - GO SET COMMAND FLAG
         B     SETJOB1                 NO  - GO SET JOB FLAG
DAYWRAP  EQU   *
         CLC   CHKPTREC(4),CURRTIME    MORE THAN ONE DAY?
         BH    WITHIN24                NO - WITHIN 24 HOURS
         MVC   CHKPTREC(4),CURRTIME    RESET TO TODAYS TIME
WITHIN24 EQU   *
         LTR   R3,R3                   CURRENT DAY IS SUNDAY?
         BZ    SATPREV                 YES - SET PREVIOUS TO SAT
         LR    R1Ø,R3                  SAVE VALUE
         S     R1Ø,=F'4'               POINT TO PREVIOUS DAY
         L     R8,DAYPTRS(R1Ø)         ADDRESS YESTERDAY'S WORK
         L     R14,DAYLEN              GET MAX DAY STORAGE
         LA    R14,Ø(R14,R8)           SET TO END OF DAY
         ST    R14,DAYEND              SAVE IT
         B     JOBLOOP2                GO CHECK FOR BACKLOG
```

```
SATPREV  L     R1Ø,=F'24'              SET TO SATURDAY
         L     R8,DAYPTRS(R1Ø)         ADDRESS YESTERDAY'S WORK
         L     R14,DAYLEN              GET MAX DAY STORAGE
         LA    R14,Ø(R14,R8)           SET TO END OF DAY
         ST    R14,DAYEND              SAVE IT
JOBLOOP2 L     R1Ø,Ø(,R8)              GET FIRST UNIT OF WORK
         LTR   R1Ø,R1Ø                 ANYTHING?
         BZ    DOTODAY                 NO - RECOVER TODAY
         C     R8,DAYEND               DONE THE DAY?
         BE    DOTODAY                 YES - RECOVER TODAY
         TM    Ø(R8),X'8Ø'             RECOVER FLAG?
         BZ    NEXTJOB2                NO - GET NEXT JOB
         CLC   Ø(2,R1Ø),CHKPTREC       BEFORE LAST CHECKPOINT?
         BNH   NEXTJOB2                YES - DON'T RECOVER
         L     R9,RECOVTBL             GET RECOVERY TBL ADDR
TBLLOOP2 CLC   Ø(8,R9),=8X'ØØ'         BLANK ENTRY?
         BE    ADDJOB2                 YES - GO ADD TO RECOV TBL
         CLC   Ø(8,R9),2(R1Ø)          SAME JOB?
         BE    CHKFLAG2                YES - GO SEE IF CMD OR JOB
         LA    R9,16(,R9)              POINT TO NEXT ENTRY
         B     TBLLOOP2                GO CHECK IT OUT
ADDJOB2  MVC   Ø(8,R9),2(R1Ø)          MOVE IN JOB
         TM    Ø(R8),X'4Ø'             A COMMAND?
         BZ    SETJOB2                 NO - SET JOB FLAG
SETCMD2  OI    8(R9),X'4Ø'             SET FLAG IN RECOVER AREA
         B     NEXTJOB2                GO CHECK NEXT ENTRY
SETJOB2  OI    8(R9),X'8Ø'             SET FLAG IN RECOVER AREA
NEXTJOB2 LA    R8,4(,R8)               POINT TO NEXT ENTRY
         B     JOBLOOP2                CHECK IT OUT
CHKFLAG2 EQU   *
         TM    8(R9),X'CØ'             BOTH FLAGS SET ALREADY?
         BO    NEXTJOB2                YES - CAN'T DO ANY MORE
         TM    Ø(R8),X'4Ø'             A COMMAND?
         BO    SETCMD2                 YES - GO SET COMMAND FLAG
         B     SETJOB2                 NO  - GO SET JOB FLAG
DOTODAY  EQU   *
         L     R8,DAYPTRS(R3)          ADDRESS CURRENT DAYS WORK
         L     R14,DAYLEN              GET MAX DAY STORAGE
         LA    R14,Ø(R14,R8)           SET TO END OF DAY
         ST    R14,DAYEND              SAVE IT
JOBLOOP3 L     R1Ø,Ø(,R8)              GET FIRST UNIT OF WORK
         LTR   R1Ø,R1Ø                 ANYTHING?
         BZ    RECOVEND                NO - NO MORE TO DO
         C     R8,DAYEND               DONE THE DAY?
         BE    RECOVEND                YES - NO MORE TO DO
         TM    Ø(R8),X'8Ø'             RECOVER FLAG?
         BZ    NEXTJOB3                NO - GET NEXT JOB
         CLC   Ø(2,R1Ø),CURRTIME       AFTER CURRENT TIME?
         BNL   RECOVEND                YES - THAT'S ALL WE NEED
         L     R9,RECOVTBL             GET RECOVERY TBL ADDR
```

```
TBLLOOP3 CLC   Ø(8,R9),=8X'ØØ'         BLANK ENTRY?
         BE    ADDJOB3                 YES - GO ADD TO RECOV TBL
         CLC   Ø(8,R9),2(R1Ø)          SAME JOB?
         BE    CHKFLAG3                YES - GO SEE IF CMD OR JOB
         LA    R9,16(,R9)              POINT TO NEXT ENTRY
         B     TBLLOOP3                GO CHECK IT OUT
ADDJOB3  MVC   Ø(8,R9),2(R1Ø)          MOVE IN JOB
         TM    Ø(R8),X'4Ø'             A COMMAND?
         BZ    SETJOB3                 NO - SET JOB FLAG
SETCMD3  OI    8(R9),X'4Ø'             SET FLAG IN RECOVER AREA
         B     NEXTJOB3                GO CHECK NEXT ENTRY
SETJOB3  OI    8(R9),X'8Ø'             SET FLAG IN RECOVER AREA
NEXTJOB3 LA    R8,4(,R8)               POINT TO NEXT ENTRY
         B     JOBLOOP3                CHECK IT OUT
CHKFLAG3 EQU   *
         TM    8(R9),X'CØ'             BOTH FLAGS SET ALREADY?
         BO    NEXTJOB3                YES - CAN'T DO ANY MORE
         TM    Ø(R8),X'4Ø'             A COMMAND?
         BO    SETCMD3                 YES - GO SET COMMAND FLAG
         B     SETJOB3                 NO  - GO SET JOB FLAG
RECOVEND EQU   *
         OPEN  JOBFILE1                OPEN JOB PDS
         TM    JOBFILE1+48,X'1Ø'       OPEN OK?
         BZ    OPENERR2                NO - ISSUE ERROR MESSAGE
         OPEN  CMDFILE1                OPEN COMMAND PDS
         TM    CMDFILE1+48,X'1Ø'       OPEN OK?
         BZ    OPENERR2                NO - ISSUE ERROR MESSAGE
         L     R9,RECOVTBL             GET MEMBER TBL ADDRESS
SUBLOOP  CLC   Ø(8,R9),=8X'ØØ'         END OF TABLE?
         BE    CLOSEIT                 YES - CLOSE DATASET
         MVC   NOMEMØ2+23(8),Ø(R9)     MOVE IN MEMBER NAME
         MVC   NOMEMØ2+19(3),=C'JOB'   MOVE IN JOB IDENTIFIER
         TM    8(R9),X'4Ø'             COMMAND FLAG SET?
         BZ    NOTCMDØ1                NO - CHECK IN JOBFILE
         FIND  CMDFILE1,(R9),D         FIND THE MEMBER
         LTR   R15,R15                 FOUND IT?
         BZ    GOSUB                   YES - GO SUBMIT IT
         MVC   NOMEMØ2+19(3),=C'CMD'   MOVE IN COMMAND IDENTIFIER
         B     NOMEMØ2                 WRITE MESSAGE
NOTCMDØ1 FIND  JOBFILE1,(R9),D         FIND THE MEMBER
         LTR   R15,R15                 FOUND IT?
         BZ    GOSUB                   YES - GO SUBMIT IT
NOMEMØ2  WTO   'SCHEDØ14 - JOB XXXXXXXX NOT FOUND, NOT SUBMITTED',     X
               ROUTCDE=(1),DESC=(1)
         B     NEXTMEM                 GET NEXT MEMBER
GOSUB    EQU   *
         OPEN  (INTRDR,OUTPUT)         OPEN INTERNAL READER
*
         TM    FLAG,HARDDOWN           HARD TERMINATION?
         BZ    READIT                  NO - DO RECOVERY
```

```
         MVC   DOUBLE(8),=C'ØØØ1ØØØØ'  SET 1 MINUTE LIMIT FOR REPLY
         L     R6,=A(REPLYEXT)         GET EXIT ADDRESS
         STIMER REAL,(R6),DINTVL=DOUBLE
         MVC   SUBWTOR+1Ø6(8),Ø(R9)    MOVE IN THE JOBNAME
         MVC   SUBWTOR+1Ø2(3),=C'JOB'  MOVE IN 'JOB'
         TM    8(R9),X'4Ø'             COMMAND FLAG SET?
         BZ    GETRIGHT                NO - DON'T MOVE IN CMD
         MVC   SUBWTOR+1Ø2(3),=C'CMD'  MOVE IN 'CMD'
GETRIGHT XC    ECBAREA(4),ECBAREA      CLEAR THE ECB
         MVI   WTOREPLY,C'Y'           SET DEFAULT REPLY
SUBWTOR  WTOR  'SCHEDØ15 - SCHEDULER CHECKPOINT SHOWS HARD TERMINATION.X
                 SHOULD RECOVERY CONTINUE FOR JOB XXXXXXXX (Y/N)?',    X
               WTOREPLY,1,ECBAREA,ROUTCDE=(1)
         WAIT  ECB=ECBAREA             WAIT FOR THE REPLY
         OI    WTOREPLY,X'4Ø'          SET TO UPPER CASE
         CLI   WTOREPLY,C'Y'           RECOVER?
         BE    READIT                  YES - DO RECOVERY
         CLI   WTOREPLY,C'N'           DON'T RECOVER?
         BE    NEXTMEM                 YES - GET NEXT ONE
         B     GETRIGHT                GET PROPER REPLY
*
READIT   LA    RØ,INPUT                GET INPUT AREA ADDRESS
         L     R1,=F'3272Ø'            GET THE LENGTH
         XR    R14,R14                 DUMMY ADDRESS
         XR    R15,R15                 MODEL BYTE
         MVCL  RØ,R14                  CLEAR THE AREA
         LA    R7,INPUT                GET AREA ADDRESS
         TM    8(R9),X'4Ø'             A COMMAND?
         BO    READCMD                 YES - GO READ THE COMMAND
         READ  DECBIN,SF,JOBFILE1,INPUT,'S' READ A BLOCK
         CHECK DECBIN                  WAIT FOR IT
PUTLOOP1 PUT   INTRDR,(R7)              WRITE A RECORD
         LA    R7,8Ø(,R7)              POINT TO NEXT RECORD
         CLC   Ø(4,R7),=F'Ø'           END OF BLOCK?
         BNE   PUTLOOP1                NO - WRITE NEXT RECORD
         B     READIT                  YES - GO FIND MORE
READCMD  READ  DECBINØ1,SF,CMDFILE1,INPUT,'S' READ A BLOCK
         CHECK DECBINØ1                WAIT FOR IT
         MODESET KEY=ZERO,MODE=SUP
CMDLOOP1 MVC   CMD+4(72),Ø(R7)         MOVE IN COMMAND
         CLI   Ø(R7),C'*'              A COMMENT?
         BE    NEXTCMD1                YES - DON'T ISSUE A COMMAND
         XR    RØ,RØ                   CLEAR RØ
         LA    R1,CMD                  GET COMMAND ADDRESS
         SVC   34                      ISSUE THE COMMAND
NEXTCMD1 LA    R7,8Ø(,R7)              POINT TO NEXT RECORD
         CLC   Ø(4,R7),=F'Ø'           END OF BLOCK?
         BNE   CMDLOOP1                NO - ISSUE NEXT COMMAND
         MODESET KEY=NZERO,MODE=PROB
         B     READIT                  GO READ NEXT BLOCK
NEXTMEM  CLOSE INTRDR                  CLOSE INTERNAL READER
         TTIMER CANCEL                 CANCEL STIMER EXIT
```

```
         TM    8(R9),X'CØ'             JOB AND CMD MEM NAME THE SAME?
         BO    CMDRESET                RESET THE COMMAND FLAG
         LA    R9,16(,R9)              POINT TO NEXT MEMBER NAME
         B     SUBLOOP                 SEE IF THERE IS MORE
CMDRESET NI    8(R9),255-X'4Ø'         RESET THE FLAG
         B     SUBLOOP                 GO BACK UP AND DO THE JOB
CLOSEIT  CLOSE JOBFILE1                CLOSE PDS
         CLOSE CMDFILE1                CLOSE PDS
         NI    FLAG,255-ACTIVE         TURN RECOVER FLAG OFF
*
SETTIMER TM    FLAG,TIMERON            TIMER SET ALREADY?
         BO    ADDTASK                 YES - GO ADD TASK
         L     R15,CHKPTINT            GET CHKPOINT INT-MINUTES
         MH    R15,=H'6ØØØ'            CONVERT TO 1/1ØØ SECONDS
         ST    R15,TIMEINT             SAVE TIME INTERVAL
         STIMERM SET,ID=TIMERID,BINTVL=TIMEINT,EXIT=CHKPTEXT
         OI    FLAG,TIMERON            SET THE FLAG
ADDTASK  EQU   *
         XC    TASKECB(4),TASKECB      CLEAR THE ECB
         ATTACH EP=SCHEDMØ2,ECB=TASKECB,SVAREA=YES,PARAM=(FLAG)
         LTR   R15,R15                 ATTACH TIMER TASK OK?
         BNZ   RETURN                  NO - GO HOME
         ST    R1,TCBADDR              SAVE TCB ADDRESS
*   SET UP ESTAE
         ESTAE SCHEDERR,CT,XCTL=NO,PARAM=ERRPARM,PURGE=NONE,ASYNCH=YES,X
               TERM=YES
*   ESTABLISH THE CONSOLE COMMUNICATION ENVIRONMENT.
         LA    R5,ANSRAREA             ADDR OF RESPONSE AREA
         EXTRACT (5),FIELDS=COMM       ADDR OF COMMUNICAT'N AREA
         L     R5,ANSRAREA             LOAD IT
         USING COMLIST,R5
         L     R3,COMCIBPT             GET ADDRESS OF CIB
         USING CIBNEXT,R3
         C     R3,=F'Ø'                CIB EXISTS?
         BE    SETCOUNT                NO - GO SET COUNT
         QEDIT ORIGIN=COMCIBPT,BLOCK=(3) YES - FREE IT
         LTR   R15,R15                 GO OK?
         BZ    SETCOUNT                YES - GO SET COUNT
REPEAT   XC    ECBAREA(4),ECBAREA      CLEAR ECB
         LA    R8,ECBAREA              GET ECB ADDRESS
         MVI   WTOREPLY,C' '           PRIME REPLY AREA
         LA    R9,WTOREPLY             GET REPLY AREA ADDRESS
         WTOR  'SCHEDØ2Ø - ERROR CONDITION RECOGNIZED IN CONSOLE INTERFX
               ACE.  PROGRAM IS TERMINATING.  SHOULD SCHEDULING REMAIN X
               ACTIVE (Y/N)?',                                         X
               (R9),1,(R8),ROUTCDE=(1)
         WAIT  ECB=ECBAREA             WAIT FOR REPLY
         OI    Ø(R9),X'4Ø'             SET TO UPPER CASE
         CLI   Ø(R9),C'Y'              LEAVE SCHEDULING ON?
         BE    WAIT                    YES - JUST GO WAIT
         CLI   Ø(R9),C'N'              TURN SCHEDULING OFF?
         BE    CSTOP                   YES - GO STOP EVERYTHING
```

```
         B     REPEAT                  RE-ISSUE MESSAGE
SETCOUNT EQU   *
*   SET LIMIT ON MODIFY COMMANDS
         QEDIT ORIGIN=COMCIBPT,CIBCTR=1  ONE MODIFY AT A TIME
         WTO   'SCHED00I - SCHEDULE CONSOLE INTERFACE ENABLED.  WAITINGX
               FOR FURTHER REQUESTS.',ROUTCDE=(1),DESC=(6)
*   WAIT FOR ANY OPERATOR REQUESTS.
WAIT     L     R8,COMECBPT             ADDR OF COMMUNICATION ECB
         ST    R8,ECBLIST              SAVE ECB ADDR IN LIST
         LA    R1,TASKECB              GET SUBTASK ECB ADDRESS
         ST    R1,ECBLIST+4            SAVE ECB ADDR IN LIST
         OI    ECBLIST+4,X'80'         SET LAST ECB FLAG
         WAIT  1,ECBLIST=ECBLIST       WAIT FOR AN EVENT
         L     R8,ECBLIST              GET FIRST ECB
         TM    0(R8),X'40'             OPER CMD EVENT COMPLETE?
         BZ    TASKERR                 NO - SUBTASK FAILED
         BAL   R6,CMDPROC              PROCESS COMMAND
         B     WAIT                    GO WAIT
*   OUR TASK HAS RECEIVED A REQUEST FROM THE OPERATOR. DETERMINE
*   THE TYPE OF REQUEST AND ACT ACCORDINGLY.
CMDPROC  EQU   *
         L     R3,COMCIBPT             GET ADDRESS OF CIB
         LTR   R3,R3                   VALID POINTER?
         BZ    RETR6                   NO - RETURN
         CLI   CIBVERB,CIBMODFY        IS IT A MODIFY COMMAND?
         BE    CMODIFY                 YES - GO PROCESS
         CLI   CIBVERB,CIBSTOP         IS IT A STOP COMMAND?
         BE    CSTOP                   YES - GO PROCESS
RETR6    EQU   *
         QEDIT ORIGIN=COMCIBPT,BLOCK=(3) FREE CIB
         BR    R6                      RETURN
CMODIFY  EQU   *
*   THIS IS A MODIFY COMMAND SO WE MUST CHECK FOR VALID SYNTAX.
         LH    R7,CIBDATLN             GET COMMAND LENGTH
         CLC   CIBDATA(4),=C'STOP'     A STOP COMMAND?
         BE    CSTOP                 YES - PROCESS LIKE STOP CMD
         B     RETR6                   GO BACK
CSTOP    EQU   *
         OI    FLAG,HALTPGM            SET HALT PROGRAM FLAG
         POST  WAITECB
TASKWAIT EQU   *
         WAIT  1,ECB=TASKECB           WAIT FOR SUBTASK TO FINISH
         DETACH TCBADDR                REMOVE THE SUBTASK
         STIMERM CANCEL,ID=TIMERID     CANCEL THE TIMER EXIT
         ESTAE 0                       CANCEL ESTAE
         OPEN  (CHKPOINT,OUTPUT)       OPEN CHECKPOINT DATASET
         TIME  DEC                     GET THE DATE AND TIME
         STM   R0,R1,CHKPTREC+8        SAVE DATE AND TIME
         PUT   CHKPOINT,CHKPTREC       WRITE OUT LAST CKPT RECORD
         CLOSE CHKPOINT                CLOSE THE DATASET
*
RETURN   L     R13,SAVE+4
```

```
         LM    R14,R12,12(R13)
         XR    R15,R15                 CLEAR RETURN CODE
         BR    R14
TASKERR  EQU   *
         WTO   'SCHED030 - AN ERROR OCCURRED IN THE SCHEDULE SUBTASK.  X
               OPERATIONS ARE TERMINATING.',ROUTCDE=(1),DESC=(1)
         B     RETURN
EXPIRED  WTO   'SCHED016 - CURRENT YEAR IS NOT SUPPORTED - PROGRAM UPDAX
               TE REQUIRED',ROUTCDE=(1),DESC=(1)
         B     RETURN
GETERR   WTO   'SCHED017 - INSUFFICIENT REGION TO FACILITATE SPECIFIED X
               JOBMAX VALUE',ROUTCDE=(1),DESC=(1)
         B     RETURN
OPENERR1 WTO   'SCHED021 - SYSIN DATASET FAILED TO OPEN, CORRECT PROBLEX
               M AND RESTART',ROUTCDE=(1),DESC=(1)
         B     RETURN
OPENERR2 WTO   'SCHED022 - JOBFILE DATASET FAILED TO OPEN, CORRECT PROBX
               LEM AND RESTART',ROUTCDE=(1),DESC=(1)
         B     RETURN
OPENERR3 WTO   'SCHED023 - CHKPOINT DATASET FAILED TO OPEN, CORRECT PROX
               BLEM AND RESTART',ROUTCDE=(1),DESC=(1)
         B     RETURN
OPENERR4 WTO   'SCHED024 - CMDFILE DATASET FAILED TO OPEN, CORRECT PROBX
               LEM AND RESTART',ROUTCDE=(1),DESC=(1)
         B     RETURN
         DS    0D
FLAG     DC    X'00'
DAILY    EQU   X'80'
CON      EQU   X'40'
WORK     DS    CL5
CURRDAY  DS    X
JULDAY   DS    CL2
RELVALUE DS    CL3
CONVALH  DS    CL3
CONVALM  DS    CL2
DOUBLE   DS    D
DATE     DS    F
TIME     DS    F
**********************
SUN      EQU   0
MON      EQU   1
TUES     EQU   2
WED      EQU   3
THUR     EQU   4
FRI      EQU   5
SAT      EQU   6
**********************
YEARTAB  DC    X'98',AL1(THUR)         1998
         DC    X'99',AL1(FRI)          1999
         DC    X'00',AL1(SAT) LEAP     2000
         DC    X'01',AL1(MON)          2001
```

```
         DC    X'Ø2',AL1(TUES)         2ØØ2
         DC    X'Ø3',AL1(WED)          2ØØ3
         DC    X'Ø4',AL1(THUR) LEAP    2ØØ4
         DC    X'Ø5',AL1(SAT)          2ØØ5
         DC    X'Ø6',AL1(SUN)          2ØØ6
         DC    X'Ø7',AL1(MON)          2ØØ7
         DC    X'Ø8',AL1(TUES) LEAP    2ØØ8
         DC    X'Ø9',AL1(THUR)         2ØØ9
         DC    X'1Ø',AL1(FRI)          2Ø1Ø
         DC    X'FF'
*
DAYTABLE DC    AL1(SUN),AL1(MON),AL1(TUES),AL1(WED),AL1(THUR),AL1(FRI)
         DC    AL1(SAT),AL1(SUN),AL1(MON),AL1(TUES),AL1(WED),AL1(THUR)
         DC    AL1(FRI),AL1(SAT)
*
NXTDPTRS DC    A(Ø),A(Ø),A(Ø),A(Ø),A(Ø),A(Ø),A(Ø)
DAYPTRS  DC    A(Ø),A(Ø),A(Ø),A(Ø),A(Ø),A(Ø),A(Ø)
*
NXTJPTRS DC    A(Ø)
JOBPTRS  DC    A(Ø)
DAYLEN   DS    F
WAITECB  DS    F
DATALEN  EQU   *-FLAG
SAVE     DS    18F
STORLIM  DS    F
SYSIN    DCB   DSORG=PS,MACRF=GL,DDNAME=SYSIN,EODAD=EOD1,LRECL=8Ø
INTRDR   DCB   DSORG=PS,MACRF=PM,DDNAME=INTRDR,LRECL=8Ø
TIMERID  DS    F
TIMEINT  DS    F
CHKPTINT DS    F
RECOVTBL DS    F
CURRTIME DS    F
CURRDATE DS    F
CHKPTREC DC    8ØC' '
CHKPOINT DCB   MACRF=(GM,PM),DSORG=PS,LRECL=8Ø,DDNAME=CHKPOINT
JOBFILE1 DCB   DSORG=PO,DDNAME=JOBFILE,EODAD=NEXTMEM,MACRF=R
CMDFILE1 DCB   DSORG=PO,DDNAME=CMDFILE,EODAD=NEXTMEM,MACRF=R
ACTIVE   EQU   X'Ø8'
TIMERON  EQU   X'Ø4'
HALTPGM  EQU   X'Ø2'
HARDDOWN EQU   X'Ø1'
TASKECB  DS    F
TCBADDR  DS    F
ECBLIST  DS    2F
ECBAREA  DS    F
WTOREPLY DS    CL1
ANSRAREA DS    F
DAYEND   DS    F
ERRPARM  DS    18F
TEMPDAT1 DS    CL8
TEMPDAT2 DS    CL8
```

```
CMD      DC    X'004C0000',80C' '
*
TRTABLE  DC    256X'80'
         ORG   TRTABLE+0
         DC    C'0123456789ABCDEF'
         ORG   TRTABLE+193
         DC    X'0A0B0C0D0E0F'
         ORG   TRTABLE+240
         DC    X'000102030405060708090'
         ORG   ,
*
         LTORG
*
INPUT    DS    CL32720
         DC    F'0'
*
PARMIN   DSECT
COMMENT  DS    0C
DAY      DS    CL9
         DS    C
TIMEID   DS    CL2
RELTIME  DS    CL4
         DS    C
JOBID    DS    CL4
JOB      DS    CL8
         DS    C
CONTID   DS    CL9
CONTIME  DS    CL4
         DS    C
RECOVID  DS    CL7
         DCBD  DSORG=PS
         CVT   DSECT=YES
         IHAPSA
*        IEFJESCT
         DSECT
         IEZCOM
         DSECT
         IEZCIB
CHKPTEXT CSECT
         STM   R14,R12,12(R13)         SAVE ENVIRONMENT
         LR    R2,R15                  SET BASE REGISTER
         USING CHKPTEXT,R2
         ST    R13,EXITSAVE+4          SAVE OLD SAVEAREA ADDRESS
         LA    R13,EXITSAVE            GET NEW SAVEAREA ADDRESS
         L     R3,=A(CHKPOINT)         GET DCB ADDRESS
         OPEN  ((R3),OUTPUT)           OPEN THE DATASET
         TIME  DEC                     GET THE TIME AND DATE
         L     R7,=A(CHKPTREC)         GET RECORD ADDRESS
         STM   R0,R1,0(R7)             SAVE IN OUTPUT AREA
         STM   R0,R1,8(R7)             SAVE IN OUTPUT AREA
         PUT   (R3),(R7)               WRITE THE RECORD
```

```
         CLOSE ((R3))                  CLOSE CHECKPOINT DATASET
         L     R4,=A(CHKPTINT)         GET ADDRESS OF INTERVAL
         L     R5,Ø(,R4)               LOAD INTERVAL VALUE
         MH    R5,=H'6ØØØ'             SET TO HUNDRETH SECONDS
         ST    R5,EXITINT              SAVE TIME INTERVAL
         STIMERM SET,ID=EXITID,BINTVL=EXITINT,EXIT=CHKPTEXT
         L     R13,EXITSAVE+4          GET OLD SAVEAREA ADDRESS
         LM    R14,R12,12(R13)         RESTORE THE ENVIRONMENT
         XR    R15,R15                 CLEAR R15
         BR    R14                     RETURN
EXITID   DS    F
EXITINT  DS    F
EXITSAVE DS    18F
         LTORG
REPLYEXT CSECT
         STM   R14,R12,12(R13)         SAVE ENVIRONMENT
         LR    R11,R15                 SET UP ...
         USING REPLYEXT,R11               EXIT ADDRESSABILITY
         ST    R13,EXITSVØ2+4
         LA    R13,EXITSVØ2
         L     R3,=A(ECBAREA)          GET ECB ADDRESS
         POST  (R3)
         L     R13,EXITSVØ2+4
         LM    R14,R12,12(R13)
         XR    R15,R15
         BR    R14
EXITSVØ2 DS    18F
         LTORG
SCHEDERR CSECT
*        PRINT NOGEN
         USING SCHEDERR,R15
         C     RØ,=F'12'                SDWA PRESENT?
         BE    NOSDWA1                  NO - PROCESS AS SUCH
         STM   R14,R12,12(R13)          SAVE ENVIRONMENT
         B     SETUP                    CONTINUE
NOSDWA1  EQU   *
         STM   R14,R12,12(R2)           R2 POINTS TO SAVE AREA PARM
         LR    R13,R2                   POINT TO SAVE AREA
SETUP    EQU   *
         DROP  R15
         LR    R11,R15                  SET UP ...
         USING SCHEDERR,R11                NEW ADDRESSABILITY
         LR    R3,RØ                    SAVE SDWA FLAG
         LR    R4,R1                    SAVE SDWA ADDRESS
         ST    R13,ERRSAVE+4            SAVE OLD SAVE AREA ADDRESS
         LA    R13,ERRSAVE              GET NEW SAVE AREA ADDRESS
         SR    R12,R12                  CLEAR R12
         C     R3,=F'12'                CHECK FOR SDWA
         BE    NOSDWA2                  NO - BYPASS SDWA PROCESSING
         LR    R12,R4                   SET UP ADDRESSABILITY ...
         USING SDWA,R12                    TO THE SDWA
```

```
         L     R2,SDWAPARM              GET PARAMETER AREA ADDRESS
NOSDWA2  EQU   *
         LTR   R12,R12                  SDWA?
         BZ    NOSDWA3                  NO - BYPASS SDWA
         UNPK  ERRDBL1(5),SDWAABCC+1(3) UNPACK ABEND CODE
         B     ERROR1
NOSDWA3  EQU   *
         ST    R4,ERRDBL2               SAVE ABEND CODE
         UNPK  ERRDBL1(5),ERRDBL2+1(3)  UNPACK ABEND CODE
ERROR1   EQU   *
         NC    ERRDBL1(6),=6X'ØF' MAKE ABEND ...
         TR    ERRDBL1(6),=C'Ø123456789ABCDEF' CODE READABLE
         CLC   ERRDBL1(3),=C'222'       OPERATOR CANCEL?
         BNE   RETDUMP                  NO - RETURN AND DUMP
         L     R6,=A(CHKPOINT)          GET DCB ADDRESS
         OPEN  ((R6),OUTPUT)            OPEN THE DATASET
         TIME  DEC                      GET THE TIME AND DATE
         L     R7,=A(CHKPTREC)          GET RECORD ADDRESS
         STM   RØ,R1,8(R7)              SAVE IN OUTPUT AREA
         PUT   (R6),(R7)                WRITE THE RECORD
         CLOSE ((R6))                   CLOSE CHECKPOINT DATASET
         L     R5,16                    GET CVT ADDRESS
         L     R5,Ø(,R5)
         L     R6,12(,R5)               CURRENT ASCB ADDRESS
         L     R7,172(,R6)              GET JOBNAME AREA ADDRESS
         LTR   R7,R7                    VALID?
         BNZ   GETJOBN                  YES - EXTRACT JOBNAME
         L     R7,176(,R6)              GET SCTNAME AREA ADDRESS
         LTR   R7,R7                    VALID?
         BZ    NONAME                   NO - DON'T FILL IN
GETJOBN  MVC   ERRWTO+19(8),Ø(R7)       MOVE IN JOBNAME
         MVC   ERRWTO+71(8),Ø(R7)       MOVE IN JOBNAME
         MVC   ERRWTO+98(8),Ø(R7)       MOVE IN JOBNAME
         MODESET KEY=ZERO,MODE=SUP
ERRWTO   WTO   'SCHEDØØ9 - SCHEDULE ACCEPTS THE ''STOP'' COMMAND.  PLEAX
               SE USE ''P SCHEDULE'' WHEN TERMINATING SCHEDULE.',      X
               ROUTCDE=(1),DESC=(1)
         MODESET KEY=NZERO,MODE=PROB
NONAME   L     R13,ERRSAVE+4            GET OLD SAVE AREA ADDRESS
         LTR   R12,R12                  SDWA?
         BZ    END                      NO - END
         SETRP WKAREA=(R4),REGS=(14),DUMP=NO,RC=Ø
RETDUMP  EQU   *
         L     R13,ERRSAVE+4            GET OLD SAVE AREA ADDRESS
         LTR   R12,R12                  SDWA?
         BZ    END                      NO - END
         SETRP WKAREA=(R4),REGS=(14),DUMP=YES,RC=Ø
END      EQU   *
         LM    RØ,R12,2Ø(R13)           RESTORE ENVIRONMENT
         XR    R15,R15                  CLEAR R15
         LA    R15,4                    SET RETRY
```

```
         L     R14,12(,R13)             POINT TO RTM
         BR    R14                      RETURN
ERRSAVE  DS    18F
ERRDBL1  DS    D
         DS    F
ERRDBL2  DS    D
         DS    F
         LTORG
         PRINT NOGEN
         IHASDWA
         END
```

SCHEDM02

```
*   ###  SUMMARY OF CHANGES
*   CHANGE IDENTIFIER       CHANGE DESCRIPTION
*   C9Ø197                  ADDED FREE TO LAST CLOSE OF SYSIN
*   C9Ø2Ø1                  ADDED SUPPORT FOR CHECKPOINTING DATASET
*                           AND RECOVER KEYWORD FOR CATCHUP PROCESSING
*
*********************************************************************
*                                                                   *
*   THE SCHEDMØ2 TASK OPERATES AS THE SCHEDULE JOB/CMD ISSUER.  IT  *
*   INTERPRETS THE TABLE OF INFORMATION BUILT BY SCHEDMØ1 TO        *
*   DETERMINE WHEN THE NEXT JOB/CMD IS TO BE SUBMITTED AND IT MANAGES *
*   THE STIMER EXIT USED TO SCHEDULE THE NEXT AVAILABLE JOB/CMD.    *
*                                                                   *
*   SCHEDMØ2 NEEDS TO RESIDE IN AN APF AUTHORIZED LIBRARY AND BE    *
*   LINKED AC(1)                                                    *
*                                                                   *
*********************************************************************
         PRINT ON,GEN
         SPACE
SCHEDMØ2 CSECT
         $SETR
         USING SCHEDMØ2,R12,R11
         STM   R14,R12,12(R13)
         LR    R12,R15                SET INITIAL BASE
         LA    R11,4Ø95(,R12)
         LA    R11,1(,R11)
         L     R9,Ø(,R1)              GET INCOMING PARMS
         USING DATAAREA,R9            SET ADDRESSABILITY
         ST    R13,SAVE+4
         LA    R13,SAVE
         LA    R1,WAITECB             GET ECB ADDRESS
         ST    R1,ECBADDR             SAVE IT
*******************************************************************
*   DAY OF WEEK CALC
GETDAY   TIME  DEC                    GET TIME AND DATE
         ST    R1,DATE                STORE DATE
```

```
         ST    RØ,TIME                 STORE TIME
*   INDEX YEAR TABLE
GETDAY2  LH    R2,DATE                 LOAD YEAR
         LA    R3,YEARTAB              ADDRESS YEAR TABLE
YEARSRCH CLM   R2,B'ØØØ1',Ø(R3)        YEAR FOUND?
         BE    YEARFND                 YES - GO PROCESS
         CLI   Ø(R3),X'FF'             NO - END OF TABLE?
         BE    EXPIRED                 YES - THE PROGRAM TABLE EXPIRED
         LA    R3,2(,R3)               NO - ADDRESS NEXT ENTRY
         B     YEARSRCH
YEARFND  SR    R2,R2                   CLEAR REGISTER
         IC    R2,1(,R3)               GET STARTING DAY OF YEAR
         XC    DOUBLE,DOUBLE           CLEAR DOUBLEWORD
         MVC   DOUBLE+6(2),DATE+2      MOVE IN DAY
         CVB   R1,DOUBLE               CONVERT DAY TO BINARY
         SR    RØ,RØ                   CLEAR EVEN REGISTER
         D     RØ,=F'7'                DIVIDE BY 7 (DAYS IN A WEEK)
         LR    R1,RØ                   MOVE REMAINDER
         S     R1,=F'1'                REMAINDER MINUS ONE
         AR    R1,R2                   PLUS STARTING DAY OF YEAR
         C     R1,=F'-1'               IS IT NEGATIVE?
         BNE   GETDAY3                 NO - JUST GO ON
         L     R1,=F'6'                SET TO SATURDAY
GETDAY3  IC    R1,DAYTABLE(R1)         GET CURRENT DAY OF THE WEEK
         STC   R1,CURRDAY              AND STORE IT
         MVC   JULDAY,DATE+2           SAVE JULIAN DAY
*****************************************************************
*   FIND FIRST JOB TO BE SUBMITTED
         SR    R3,R3                   CLEAR REGISTER
         IC    R3,CURRDAY              GET CURRENT DAY
         MH    R3,=H'4'                GET DAY POINTER OFFSET
         L     R2,DAYPTRS(R3)          ADDRESS TODAYS WORK
         L     R14,DAYLEN              GET MAX DAY STORAGE
         LA    R14,Ø(R14,R2)           SET TO END OF DAY
         ST    R14,DAYEND              SAVE IT
         L     R15,DAYTBL(R3)
         MVC   STATWTO+19(9),Ø(R15)
         ST    R2,DBL2
         UNPK  DBL1(9),DBL2(5)
         NC    DBL1(8),=8X'ØF'
         TR    DBL1(8),=C'Ø123456789ABCDEF'
         MVC   STATWTO+43(8),DBL1
         ST    R14,DBL2
         UNPK  DBL1(9),DBL2(5)
         NC    DBL1(8),=8X'ØF'
         TR    DBL1(8),=C'Ø123456789ABCDEF'
         MVC   STATWTO+63(8),DBL1
STATWTO  WTO   'SCHEDØØ2 - XXXXXXXXX:  DAY START - XXXXXXXX  DAY END - X
               XXXXXXXX',ROUTCDE=(1),DESC=(6)
JOBSRCH  EQU   *
         L     R3,Ø(,R2)               ADDRESS JOB
```

```
         LTR   R3,R3                   ANY WORK FOR TODAY?
         BZ    DAYSLEEP                NO - SLEEP FOR THE DAY
         CLC   Ø(2,R3),TIME            JOB FOR THE FUTURE?
         BE    SUBMIT                  NO - SUBMIT IT RIGHT NOW
         BH    WAITTIME                YES - GO SET UP
         LA    R2,4(,R2)               NO - ADDRESS NEXT JOB
         B     JOBSRCH                 AND GO CHECK IT OUT
*****************************************************************
*   R2 POINTS TO JOB ADDRESS
*   R3 POINTS TO JOB
WAITTIME UNPK  DOUBLE(5),Ø(3,R3)       UNPACK JOB SUBMIT TIME
         MVC   DOUBLE+4(4),=C'ØØØØ'    FILL IN REMAINDER
         L     R8,=A(TIMEEXIT)         GET TIMER EXIT ADDRESS
         STIMER REAL,(R8),TOD=DOUBLE   WAIT UNTIL SUBMIT TIME
         XC    WAITECB(4),WAITECB      CLEAR THE ECB
         LA    R8,WAITECB              GET ECB ADDRESS
         WAIT  1,ECB=(R8)              WAIT FOR THE TIME TO EXPIRE
         TM    FLAG,HALTPGM            PROGRAM HAS BEEN HALTED?
         BZ    NORETØ1                 NO - DON'T GO BACK
         TTIMER CANCEL                 CANCEL THE TIMER EXIT
         L     R15,=F'4'               SET RETURN CODE 4
         B     NONZERO                 RETURN NONZERO
NORETØ1  EQU   *
*****************************************************************
SUBMIT   EQU   *
         TM    Ø(R2),X'4Ø'             A COMMAND?
         BO    CMDØ1                   YES - DO COMMAND STUFF
         OPEN  (JOBFILE)               OPEN THE PDS
         LA    RIØ,JOBFILE             ADDRESS DCB
         USING IHADCB,RIØ
         TM    DCBOFLGS,X'1Ø'          OPEN OK?
         BZ    OPENERR2                NO - INDCIATE ERROR
         DROP  RIØ
         B     FINDØ1                  GO FIND MEMBER
CMDØ1    EQU   *
         OPEN  (CMDFILE)               OPEN THE PDS
         TM    CMDFILE+48,X'1Ø'        OPEN OK?
         BZ    OPENERR3                NO - ISSUE ERROR
         B     FINDØ2
FINDØ1   LA    R4,2(,R3)               ADDRESS THE MEMBER NAME
         FIND  JOBFILE,(R4),D          LOCATE THE MEMBER
         LTR   R15,R15                 FIND GO OK?
         BE    GETRDR                  YES - CONTINUE
FINDERR  MVC   NOMEM+23(8),2(R3)       NO - SET UP AND ISSUE MESSAGE
         MVC   NOMEM+19(3),=C'JOB'     MOVE IN IDENTIFIER
NOMEM    WTO   'SCHEDØ14 - JOB XXXXXXXX NOT FOUND, NOT SUBMITTED',     X
               ROUTCDE=(1),DESC=(1)
         B     SUBEND
FINDØ2   LA    R4,2(,R3)               ADDRESS THE MEMBER NAME
         FIND  CMDFILE,(R4),D          LOCATE THE MEMBER
```

```
         LTR   R15,R15                 FIND GO OK?
         BE    GETRDR                  YES - CONTINUE
         MVC   NOMEM+19(3),=C'CMD'     MOVE IN IDENTIFIER
         B     FINDERR                 ISSUE ERROR
*****************************************************************
GETRDR   EQU  *
         TM    Ø(R2),X'4Ø'             A COMMAND?
         BZ    OPENRDR                 NO - OPEN INTERNAL READER
READMEM2 LA    R4,INPUT                ADDRESS STORAGE
         L     R5,=F'32720'            GET LENGTH
         SR    R6,R6                   DUMMY ADDRESS
         SR    R7,R7                   ZERO SECOND LENGTH
         MVCL  R4,R6                   CLEAR AREA
         READ  DECBINØ1,SF,CMDFILE,INPUT,'S' READ A RECORD
         CHECK DECBINØ1                WAIT FOR THE RECORD
         LA    R7,INPUT                ADDRESS INPUT
         MODESET KEY=ZERO,MODE=SUP
CMDLOOP  MVC   CMD+4(72),Ø(R7)         MOVE IN COMMAND
         CLI   Ø(R7),C'*'              A COMMENT?
         BE    NEXTCMD                 YES - DON'T ISSUE COMMAND
         XR    RØ,RØ                   CLEAR RØ
         LA    R1,CMD                  GET COMMAND ADDRESS
         SVC   34                      ISSUE THE COMMAND
NEXTCMD  LA    R7,8Ø(,R7)              ADDRESS NEXT RECORD
         CLC   Ø(4,R7),=F'Ø'           RECORD EXIST?
         BNE   CMDLOOP                 YES - GO ISSUE COMMAND
         MODESET KEY=NZERO,MODE=PROB
         B     READMEM2                NO - GO GET SOME MORE
PDSEODØ2 CLOSE CMDFILE
         B     SUBEND
OPENRDR  OPEN (INTRDR,OUTPUT)          OPEN INTRDR
READMEM  LA    R4,INPUT                ADDRESS STORAGE
         L     R5,=F'32720'            GET LENGTH
         SR    R6,R6                   DUMMY ADDRESS
         SR    R7,R7                   ZERO SECOND LENGTH
         MVCL  R4,R6                   CLEAR AREA
         READ  DECB,SF,JOBFILE,INPUT,'S' READ A RECORD
         CHECK DECB                    WAIT FOR THE RECORD
         LA    R7,INPUT                ADDRESS INPUT
PUTLOOP  PUT   INTRDR,(R7)              WRITE RECORD TO INTRDR
         LA    R7,8Ø(,R7)              ADDRESS NEXT RECORD
         CLC   Ø(4,R7),=F'Ø'           RECORD EXIST?
         BNE   PUTLOOP                 YES - GO WRITE IT OUT
         B     READMEM                 NO - GO GET SOME MORE
PDSEOD   CLOSE (INTRDR,,JOBFILE)
*****************************************************************
SUBEND   TIME  DEC
         ST    R1,DATE                 STORE DATE
         ST    RØ,TIME                 STORE TIME
         LA    R2,4(,R2)               ADDRESS NEXT JOB
         C     R2,DAYEND               END OF DAY?
```

```
         BE    DAYCHECK                YES - CHECK FOR SAME DAY
         L     R3,Ø(,R2)               ADDRESS JOB
         LTR   R3,R3                   ANY MORE WORK FOR TODAY?
         BZ    DAYCHECK                NO - CHECK FOR SAME DAY
         CLC   Ø(2,R3),TIME            JOB SHOULD HAVE BEEN SUBMITTED?
         BNH   SUBMIT                  YES - GO SUBMIT IT
         B     WAITTIME                NO - THEN GO SET UP
DAYCHECK CLC   DATE+2(2),JULDAY        STILL IN THE SAME DAY?
         BE    DAYSLEEP                YES - THEN SLEEP FOR THE REST
         B     NEWDAY                  NO - THEN START OVER...NEW DAY
*****************************************************************
DAYSLEEP MVC   DOUBLE,=C'23595ØØ'     WAKE UP AT MIDNIGHT
         L     R8,=A(TIMEEXIT)         GET TIMER EXIT ADDRESS
         STIMER REAL,(R8),TOD=DOUBLE   WAIT UNTIL SUBMIT TIME
         XC    WAITECB(4),WAITECB      CLEAR THE ECB
         LA    R8,WAITECB              GET ECB ADDRESS
         WAIT  1,ECB=(R8)              WAIT FOR THE TIME TO EXPIRE
         TM    FLAG,HALTPGM            PROGRAM HAS BEEN HALTED?
         BZ    NORETØ2                 NO - DON'T GO BACK
         TTIMER CANCEL
         L     R15,=F'8'               SET RETURN CODE 8
         B     NONZERO                 RETURN NONZERO
NORETØ2  EQU   *
         MVC   DOUBLE,=C'ØØØ1ØØØØ'     ONE MINTUE
         L     R8,=A(TIMEEXIT)         GET TIMER EXIT ADDRESS
         STIMER REAL,(R8),DINTVL=DOUBLE   WAIT UNTIL SUBMIT TIME
         XC    WAITECB(4),WAITECB      CLEAR THE ECB
         LA    R8,WAITECB              GET ECB ADDRESS
         WAIT  1,ECB=(R8)              WAIT FOR THE TIME TO EXPIRE
         TM    FLAG,HALTPGM            PROGRAM HAS BEEN HALTED?
         BZ    NORETØ3                 NO - DON'T GO BACK
         TTIMER CANCEL
         L     R15,=F'12'              SET RETURN CODE 12
         B     NONZERO                 RETURN NONZERO
NORETØ3  EQU   *
NEWDAY   TIME  DEC
         ST    R1,DATE                 STORE DATE
         MVC   TIME,=X'ØØØØØ1ØØ'       SET FOR FIRST SECOND OF NEW DAY
         B     GETDAY2                 GO CALCULATE THE DAY OF THE WEEK
*****************************************************************
RETURN   L     R13,SAVE+4              RESTORE SAVEAREA
         LM    R14,R12,12(R13)         RESTORE ENVIRONMENT
         XR    R15,R15                 SET RETURN CODE
         BR    R14
NONZERO  L     R13,SAVE+4
         L     R14,12(,R13)
         LM    RØ,R12,2Ø(R13)
         BR    R14
EXPIRED  WTO   'SCHEDØ16 - CURRENT YEAR IS NOT SUPPORTED - PROGRAM UPDAX
               TE REQUIRED',ROUTCDE=(1),DESC=(1)
         B     RETURN
```

```
OPENERR2 WTO   'SCHED022 - JOBFILE DATASET FAILED TO OPEN, CORRECT PROBX
               LEM AND RESTART',ROUTCDE=(1),DESC=(1)
         B     RETURN
OPENERR3 WTO   'SCHED024 - CMDFILE DATASET FAILED TO OPEN, CORRECT PROBX
               LEM AND RESTART',ROUTCDE=(1),DESC=(1)
         B     RETURN
DAYTBL   DC    A(SUNDAY)
         DC    A(MONDAY)
         DC    A(TUESDAY)
         DC    A(WEDNESDAY)
         DC    A(THURSDAY)
         DC    A(FRIDAY)
         DC    A(SATURDAY)
SUNDAY   DC    C'SUNDAY   '
MONDAY   DC    C'MONDAY   '
TUESDAY  DC    C'TUESDAY  '
WEDNESDAY DC   C'WEDNESDAY'
THURSDAY DC    C'THURSDAY '
FRIDAY   DC    C'FRIDAY   '
SATURDAY DC    C'SATURDAY '
DBL1     DS    2D
DBL2     DS    2D
SAVE     DS    18F
STORLIM  DS    F
DAYEND   DS    F
ECBADDR  DS    F
CMD      DC    X'004C0000',80C' '
INTRDR   DCB   DSORG=PS,MACRF=PM,DDNAME=INTRDR,LRECL=80
JOBFILE  DCB   DSORG=PO,DDNAME=JOBFILE,EODAD=PDSEOD,MACRF=R
CMDFILE  DCB   DSORG=PO,DDNAME=CMDFILE,EODAD=PDSEOD02,MACRF=R
         LTORG
INPUT    DS    CL3272Ø
         DC    F'Ø'
DATAAREA DSECT
         DS    ØD
FLAG     DS    XL1
DAILY    EQU   X'8Ø'
CON      EQU   X'4Ø'
HALTPGM  EQU   X'Ø2'
WORK     DS    CL5
CURRDAY  DS    X
JULDAY   DS    CL2
RELVALUE DS    CL3
CONVALH  DS    CL3
CONVALM  DS    CL2
DOUBLE   DS    D
DATE     DS    F
TIME     DS    F
**********************
SUN      EQU   Ø
```

```
MON      EQU   1
TUES     EQU   2
WED      EQU   3
THUR     EQU   4
FRI      EQU   5
SAT      EQU   6
**********************
YEARTAB  DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1,AL1
         DS    XL1
DAYTABLE DS    AL1,AL1,AL1,AL1,AL1,AL1
         DS    AL1,AL1,AL1,AL1,AL1,AL1
         DS    AL1,AL1
NXTDPTRS DS    A,A,A,A,A,A,A
DAYPTRS  DS    A,A,A,A,A,A,A
NXTJPTRS DS    A
JOBPTRS  DS    A
DAYLEN   DS    F
WAITECB  DS    F
DATALEN  EQU   *-DATAAREA
         DCBD  DSORG=PS
TIMEEXIT CSECT
         STM   R14,R12,12(R13)       SAVE ENVIRONMENT
         LR    R11,R15               SET UP ...
         USING TIMEEXIT,R11             EXIT ADDRESSABILITY
         ST    R13,EXITSVØ2+4
         LA    R13,EXITSVØ2
         L     R3,=A(ECBADDR)        GET ADDRESS OF ECB ADDRESS
         L     R4,Ø(,R3)             GET ECB ADDRESS
         POST  (R4)
         L     R13,EXITSVØ2+4
         LM    R14,R12,12(R13)
         XR    R15,R15
         BR    R14
EXITSVØ2 DS    18F
         LTORG
         END
```


Assembler instruction trace – part 2

This month we continue our look at the code for the Assembler instruction trace.

```
      ENDIF
   ELSE
      IC    R14,XCELL
      A     R14,=A(AR_ØØ)
      IF    IAC,RØ,NZ
         SELECT
         WHEN  TM,FLAGS+1,RXBIT+RSBIT,NZ
            IF    TM,Ø(R14),AR_B2,O
               LAM   R1,R1,Ø(R15)
            ENDIF
         WHEN  TM,FLAGS+1,SIBIT,O
            IF    TM,Ø(R14),AR_B1,O
               LAM   R1,R1,Ø(R15)
            ENDIF
         WHEN  TM,FLAGS+1,SSBIT,O
            LA    RØ,XCELL+2
            SELECT
            WHEN  CR,RØ,EQ,R8
               IF    TM,Ø(R14),AR_B1,O
                  LAM   R1,R1,Ø(R15)
               ENDIF
            WHEN  TM,Ø(R14),AR_B2,O
               LAM   R1,R1,Ø(R15)
            ENDSEL
         ENDSEL
      ENDIF
   ENDIF
   IF    LTR,R5,R5,NZ
      IF    CLI,XCELL,EQ,X'DB'
         LA    R1,XMS_WRK
      ENDIF
      LR    R3,R5
      DO    WHILE=(C,R3,GT,=F'7')
         UNPK  Ø(15,R2),Ø(8,R1)
         LA    R2,14(,R2)
         LA    R1,7(,R1)
         S     R3,=F'7'
      ENDDO
      LR    R14,R3
      BCTR  R14,Ø
      EX    R14,MOVE_OP
      LR    R14,R3
      SLL   R14,1+4           .*2, AND SHIFT TO NEXT NIBBLE
      LA    R15,Ø(R3,R14)
```

```
            EX    R15,UNPK_OP
            SLL   R3,1
            LA    R14,Ø(R3,R2)
            MVI   Ø(R14),X'4Ø'
            SLL   R5,1                    .* 2
            BCTR  R5,Ø                    .MAKE EXEC LEN
            EX    R5,TRANS
         ENDIF
         CPYA   R1,R12
         MODEXIT
         EJECT
SHOW_GRS MODENTRY
         LR    R1,R3
         N     R1,=A(X'FØ')
         SRL   R1,2
         IF    CLC,=X'B24D',EQ,XCELL,  .CPYA OR SAR?                  +
               OR,CLC,=X'B24E',EQ,XCELL
            LA    R1,AR_SAVE(R1)       .FIRST REG IS ACCESS REG
         ELSE
            LA    R1,REGTBL(R1)
         ENDIF
         UNPK  GR_1(9),Ø(5,R1)
         MVI   GR_1+8,X'4Ø'
         TR    GR_1(8),HEXCHAR-C'Ø'
         IF    TM,FLAGS,DBLBIT,O,OR,CLI,XCELL,EQ,X'DD'
            UNPK  DR1(9),4(5,R1)
            MVI   DR1+8,X'4Ø'
            TR    DR1(8),HEXCHAR-C'Ø'
         ENDIF
         SELECT
         WHEN  CLI,XCELL,EQ,X'B2'
            IF    TM,FLAGS+1,B2R2BIT,O
               PERF  SHOW_GR2
            ENDIF
         WHEN  TM,FLAGS+1,RRBIT,O,AND,CLI,XCELL,NE,4,ORIF,            +
               CLI,XCELL,EQ,X'A8',OR,CLI,XCELL,EQ,X'A9',OR,           +
               CLI,XCELL,EQ,X'86',OR,CLI,XCELL,EQ,X'BA',OR,           +
               CLI,XCELL,EQ,X'87'
            PERF  SHOW_GR2
         ENDSEL
         MODEXIT
         EJECT
SHOW_GR2 MODENTRY
         LR    R1,R3
         N     R1,=F'15'
         SLL   R1,2
         SELECT
         WHEN  CLC,=X'B24D',EQ,XCELL,   .CPYA OR EAR?                 +
               OR,CLC,=X'B24F',EQ,XCELL
            LA    R1,AR_SAVE(R1)       .SECOND REG IS ACCESS REG
         WHEN  CLI,XCELL,EQ,5,ORIF,CLI,XCELL,GE,X'ØB',AND,            +
               CLI,XCELL,LE,X'ØD'
```

```
            LA    R1,OLDREGS(R1)
         WHEN  NONE
            LA    R1,REGTBL(R1)
         ENDSEL
         UNPK  GR_2(9),Ø(5,R1)
         MVI   GR_2+8,X'4Ø'
         TR    GR_2(8),HEXCHAR-C'Ø'
         SELECT
         WHEN  CLI,XCELL,EQ,X'86',OR,CLI,XCELL,EQ,X'87',ANDIF,         +
               TM,XCELL+1,1,Z
            UNPK  DR2B(9),4(5,R1)
            MVI   DR2B+8,X'4Ø'
            TR    DR2B(8),HEXCHAR-C'Ø'
         WHEN  TM,FLAGS,DBLBIT,O,AND,CLI,XCELL,NE,X'1C',               +
               AND,CLI,XCELL,NE,X'1D'
            UNPK  DR2B(9),4(5,R1)
            MVI   DR2B+8,X'4Ø'
            TR    DR2B(8),HEXCHAR-C'Ø'
         ENDSEL
         MODEXIT
         EJECT
DUMPREGS MODENTRY
         PERF  WRITE
         LA    R3,REGTBL
         LA    R4,PRTLINE+5
         DO    FROM=(R5,Ø),BY=(R14,1),TO=(R15,7)
            MVC   Ø(4,R4),=C'RØØ='
            CVD   R5,DUB
            OI    DUB+7,X'ØF'
            UNPK  1(2,R4),DUB+6(2)
            UNPK  4(9,R4),Ø(5,R3)
            MVI   12(R4),X'4Ø'
            TR    4(8,R4),HEXCHAR-C'Ø'
            LA    R3,4(,R3)
            LA    R4,15(,R4)
         ENDDO
         PERF  WRITE
         LA    R3,REGTBL+8*4
         LA    R4,PRTLINE+5
         DO    FROM=(R5,8),BY=(R14,1),TO=(R15,15)
            MVC   Ø(4,R4),=C'RØØ='
            CVD   R5,DUB
            OI    DUB+7,X'ØF'
            UNPK  1(2,R4),DUB+6(2)
            UNPK  4(9,R4),Ø(5,R3)
            MVI   12(R4),X'4Ø'
            TR    4(8,R4),HEXCHAR-C'Ø'
            LA    R3,4(,R3)
            LA    R4,15(,R4)
         ENDDO
         PERF  WRITE
         PERF  WRITE
```

```
         MODEXIT
         EJECT
DUMP_ARS MODENTRY
         LA    R3,AR_SAVE
         LA    R4,PRTLINE+4
         DO    FROM=(R5,Ø),BY=(R14,1),TO=(R15,7)
            MVC   Ø(5,R4),=C'ARØØ='
            CVD   R5,DUB
            OI    DUB+7,X'ØF'
            UNPK  2(2,R4),DUB+6(2)
            UNPK  5(9,R4),Ø(5,R3)
            MVI   13(R4),X'4Ø'
            TR    5(8,R4),HEXCHAR-C'Ø'
            LA    R3,4(,R3)
            LA    R4,15(,R4)
         ENDDO
         PERF  WRITE
         LA    R3,AR_SAVE+8*4
         LA    R4,PRTLINE+4
         DO    FROM=(R5,8),BY=(R14,1),TO=(R15,15)
            MVC   Ø(5,R4),=C'ARØØ='
            CVD   R5,DUB
            OI    DUB+7,X'ØF'
            UNPK  2(2,R4),DUB+6(2)
            UNPK  5(9,R4),Ø(5,R3)
            MVI   13(R4),X'4Ø'
            TR    5(8,R4),HEXCHAR-C'Ø'
            LA    R3,4(,R3)
            LA    R4,15(,R4)
         ENDDO
         PERF  WRITE
         PERF  WRITE
         MODEXIT
         EJECT
DUMP_FLT MODENTRY
         LA    R3,PRTLINE+5
         LA    R4,FLTRØ
         DO    FROM=(R5,Ø),BY=(R14,2),TO=(R15,6)
            MVC   Ø(4,R3),=C'FRØ='
            CVD   R5,DUB
            OI    DUB+7,X'ØF'
            UNPK  2(1,R3),DUB+7(1)
            UNPK  4(9,R3),Ø(5,R4)
            UNPK  12(9,R3),4(5,R4)
            MVI   2Ø(R3),X'4Ø'
            TR    4(16,R3),HEXCHAR-C'Ø'
            LA    R4,8(,R4)
            LA    R3,3Ø(,R3)
         ENDDO
         PERF  WRITE
         PERF  WRITE
         MODEXIT
```

```
SHOW_AR  MODENTRY
         IC    R1,Ø(,R8)
         N     R1,=A(X'ØØØØØØFØ')
         MVC   Ø(5,R6),=C'ARXX='
         SRL   R1,4
         CVD   R1,DUB
         OI    DUB+7,X'ØF'
         UNPK  2(2,R6),DUB+6(2)
         SLL   R1,2
         LA    R1,Ø(R1,R5)
         UNPK  5(9,R6),Ø(5,R1)
         MVI   13(R6),X'4Ø'
         TR    5(8,R6),HEXCHAR-C'Ø'
         MODEXIT
         EJECT
         EJECT
CLEANUP  MODENTRY
         PERF  DUMPREGS
         PERF  DUMP_FLT
         MVC   PRTLINE(26),=C'     RESUME EXECUTION AT: '
         UNPK  PRTLINE+26(9),REGTBL+14*4(5)
         MVI   PRTLINE+26+8,X'4Ø'
         TR    PRTLINE+26(8),HEXCHAR-C'Ø'
         PERF  WRITE
         MVC   PRTLINE+1(133),=133C'='
         PERF  WRITE
         PERF  KILLXMS
         STM   R2,R4,XMSSTAT
         CLOSE ACB,MF=(E,CLOSELST),MODE=31
         DLVRP MODE=31,MF=(E,DLVRP)
         LM    R2,R4,XMSSTAT
         PERF  RSETXMS
         MODEXIT
         EJECT
PLO_PRIME_REGS MODENTRY
         XR    R1,R1
         IC    R1,XCELL+1
         LR    R15,R1
         IF    N,R1,=A(X'FØ'),NZ
            SRL   R1,4-2
            LA    R1,REGTBL(R1)
            IF    TM,XCELL+1,X'1Ø',O
               L     R3,Ø(,R1)
               OI    CODEFLD+1,X'3Ø'
            ELSE
               LM    R2,R3,Ø(R1)
               OI    CODEFLD+1,X'2Ø'
            ENDIF
         ENDIF
         IF    N,R15,=A(X'ØF'),NZ
            SLL   R15,2
            LA    R1,REGTBL(R15)
```

```
            IF    TM,XCELL+1,X'Ø1',O
               L     R5,Ø(,R1)
               OI    CODEFLD+1,X'Ø5'
            ELSE
               LM    R4,R5,Ø(R1)
               OI    CODEFLD+1,X'Ø4'
            ENDIF
         ENDIF
         MODEXIT
         TITLE 'ILLEGAL OP-CODE, OR EX OF AN EX INSTR'
ILGLOP   DS    ØH
         LAM   RØ,R15,=16F'Ø'
         PERF  SHOWINST
         PERF  WRITE
         IF    CLI,XCELL,EQ,X'44'
            LA    R15,X'ØC3'
         ELSE
            LA    R15,X'ØC1'
         ENDIF
         ABEND (R15),DUMP,,SYSTEM
         EJECT
         TITLE '****************** C O N S T A N T S *****************'
         DS    ØD
MOVE_LIT MVC   8(Ø,R3),5(R5)
ORI      OI    XCELL+1,Ø
TRANS    TR    11(Ø,R6),HEXCHAR-C'Ø'
MOVE_OP  MVC   DUB(Ø),Ø(R1)
UNPK_OP  UNPK  Ø(Ø,R2),DUB(Ø)
HEXCHAR  DC    C'Ø123456789ABCDEF'
BCDCC    DC    C'8421'
HEXCC    DC    X'Ø8Ø4Ø2Ø1'
         TITLE 'SUB-ROUTINES, WITH POSSIBLE VARYING BASE'
EXEC_SVC MODENTRY  NEWBASE=R1Ø
         SELECT
         WHEN  CLI,XCELL+1,EQ,3
            MVC   PRTLINE(63),=C'****** TRACE TERMINATED BY EXECUTION O+
               F SVC 3 (EXIT SVC) ******'
            PERF  WRITE
         WHEN  CLI,XCELL+1,EQ,7
            MVC   PRTLINE(63),=C'****** TRACE TERMINATED BY EXECUTION O+
               F SVC 7 (XCTL/XCTLX) ****'
            PERF  WRITE
         WHEN  CLI,XCELL+1,EQ,55
            MVC   PRTLINE(63),=C'****** TRACE TERMINATED BY EXECUTION O+
               F SVC 55 (EOV SVC) ******'
            PERF  WRITE
         ENDSEL
         PERF  WRITE
         MVC   PRTLINE(4Ø),=C' ****** REGISTERS BEFORE SVC ØØØ ****** '
         XR    RØ,RØ
         IC    RØ,CODEFLD+1
         CVD   RØ,DUB
```

```
OI    DUB+7,X'ØF'
UNPK  PRTLINE+29(3),DUB+6(2)
IF    CLI,CODEFLD+1,EQ,7,OR,CLI,CODEFLD+1,EQ,3,OR,            +
      CLI,CODEFLD+1,EQ,55
   ST    R9,REGTBL+14*4
   B     BREAK_LOOP
ENDIF
PERF  DUMPREGS
SELECT
WHEN  CLI,CODEFLD+1,EQ,7
   LAM   RØ,R15,AR_SAVE
   LM    RØ,R15,REGTBL
   SVC   7
WHEN  CLI,CODEFLD+1,EQ,3
   LAM   RØ,R15,AR_SAVE
   LM    RØ,R15,REGTBL
   SVC   3
WHEN  CLI,CODEFLD+1,EQ,55
   LAM   RØ,R15,AR_SAVE
   LM    RØ,R15,REGTBL
   SVC   55
ENDSEL
LM    RØ,R2,REGTBL            .RESTORE ALL REGS THAT MAY BE
LM    R13,R15,REGTBL+13*4     .USED BY SVC
LAM   RØ,R6,AR_SAVE
LAM   R8,R15,AR_SAVE+8*4
EX    Ø,CODEFLD
STAM  RØ,R6,AR_SAVE
STAM  R8,R15,AR_SAVE+8*4
LAM   RØ,R15,=16F'Ø'
STM   RØ,R2,REGTBL            .AND SAVE SAME, IN CASE THEY
STM   R13,R15,REGTBL+13*4     .WERE MODIFIED BY SVC
LR    R13,R7                  .POINT R13 AT MY SAVE AREA
PERF  SHOWINST
XR    R1,R1
IC    R1,XCELL+1              .SHOW SVC NUMBER IN DECIMAL
CVD   R1,DUB
OI    DUB+7,X'ØF'
UNPK  FIELDS(3),DUB+6(2)
IF    C,R1,GT,=F'199'
   MVC   FIELDS+15(4Ø),=CL4Ø'USER SVC'
ELSE
   MH    R1,=H'4Ø'           .AND DISPLAY THE RELATED MACRO
   L     R15,=A(SVCNAMES)
   LA    R1,Ø(R1,R15)
   MVC   FIELDS+15(4Ø),Ø(R1)
   PERF  EXTRA_SVC_INFO
ENDIF
L     R9,NEW_IPTR
LA    R9,2(,R9)
MODEXIT
EJECT
```

```
EXTRA_SVC_INFO MODENTRY
         XR    R1,R1
         IC    R1,CODEFLD+1
         SLL   R1,2
         L     R15,SVC_INFO_PROCESSORS(R1)
         BASR  R14,R15
*        BAL   R1,SVC_INFO_PROCESSORS(R1)
*        B     AROUND_PROCESSORS
AROUND_PROCESSORS DS ØH
         MODEXIT
SVC_INFO_PROCESSORS   DS ØH
         DC    A(AROUND_PROCESSORS)         .SVC ØØØ, EXCP/XDAP
         DC    A(INFO_WAIT)                 .SVC Ø01, WAIT
         DC    A(INFO_POST)                 .SVC Ø02, POST
         DC    A(AROUND_PROCESSORS)         .SVC Ø03, EXIT
         DC    A(AROUND_PROCESSORS)         .SVC Ø04, GETMAIN
         DC    A(AROUND_PROCESSORS)         .SVC Ø05, FREEMAIN
         DC    A(INFO_LINK_LOAD)            .SVC Ø06, LINK(X)
         DC    A(AROUND_PROCESSORS)         .SVC Ø07, XCTL(X)
         DC    A(INFO_LINK_LOAD)            .SVC Ø08, LOAD(X)
         DC    A(INFO_LINK_LOAD)            .SVC Ø09, DELETE
         DC    A(INFO_SVC1Ø)                .SVC Ø1Ø, GETMAIN/FREEMAIN
         DC    A(AROUND_PROCESSORS)         .SVC Ø11, TIME
         DC    A(AROUND_PROCESSORS)         .SVC Ø12, SYNCH(X)
         DC    A(AROUND_PROCESSORS)         .SVC Ø13, ABEND
         DC    A(AROUND_PROCESSORS)         .SVC Ø14, SPIE
         DC    A(AROUND_PROCESSORS)         .SVC Ø15, ERREXCP
         DC    A(AROUND_PROCESSORS)         .SVC Ø16, PURGE
         DC    A(AROUND_PROCESSORS)         .SVC Ø17, RESTORE
         DC    A(INFO_BLDL_FIND)            .SVC Ø18, BLDL/FIND
         DC    A(INFO_OPEN_CLOSE)           .SVC Ø19, OPEN
         DC    A(INFO_OPEN_CLOSE)           .SVC Ø2Ø, CLOSE
         DC    A(INFO_STOW)                 .SVC Ø21, STOW
         DC    A(AROUND_PROCESSORS)         .SVC Ø22, OPEN TYPE=J
         DC    A(AROUND_PROCESSORS)         .SVC Ø23, CLOSE TYPE=T
         DC    A(AROUND_PROCESSORS)         .SVC Ø24, DEVTYPE
         DC    A(AROUND_PROCESSORS)         .SVC Ø25, TRKBAL
         DC    A(AROUND_PROCESSORS)         .SVC Ø26, CALTG/INDX/LOCAT
         DC    A(AROUND_PROCESSORS)         .SVC Ø27, OBTAIN
         DC    A(AROUND_PROCESSORS)         .SVC Ø28, RESERVED
         DC    A(AROUND_PROCESSORS)         .SVC Ø29, SCRATCH
         DC    A(AROUND_PROCESSORS)         .SVC Ø3Ø, RENAME
         DC    A(AROUND_PROCESSORS)         .SVC Ø31, FEOV
         DC    A(AROUND_PROCESSORS)         .SVC Ø32, REALLOC
         DC    A(AROUND_PROCESSORS)         .SVC Ø33, IOHALT
         DC    A(AROUND_PROCESSORS)         .SVC Ø34, MGCR(E)/QEDIT
         DC    A(AROUND_PROCESSORS)         .SVC Ø35, WTO(R)
         DC    A(AROUND_PROCESSORS)         .SVC Ø36, WTL
         DC    A(AROUND_PROCESSORS)         .SVC Ø37, SEGLD/SEGWT
         DC    A(AROUND_PROCESSORS)         .SVC Ø38, RSVD
         DC    A(AROUND_PROCESSORS)         .SVC Ø39, LABEL
         DC    A(AROUND_PROCESSORS)         .SVC Ø4Ø, EXTRACT
```

```
DC    A(AROUND_PROCESSORS)         .SVC Ø41, IDENTIFY
DC    A(INFO_ATTACH)               .SVC Ø42, ATTACH(X)
DC    A(AROUND_PROCESSORS)         .SVC Ø43, CIRB
DC    A(AROUND_PROCESSORS)         .SVC Ø44, CHAP
DC    A(AROUND_PROCESSORS)         .SVC Ø45, OVLYBRCH
DC    A(AROUND_PROCESSORS)         .SVC Ø46, TTIMER/STIMERM
DC    A(AROUND_PROCESSORS)         .SVC Ø47, STIMER(M)
DC    A(AROUND_PROCESSORS)         .SVC Ø48, DEQ
DC    A(AROUND_PROCESSORS)         .SVC Ø49, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø5Ø, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø51, SNAP(X)/SDUMP(X)
DC    A(AROUND_PROCESSORS)         .SVC Ø52, RESTART
DC    A(AROUND_PROCESSORS)         .SVC Ø53, RELEX
DC    A(AROUND_PROCESSORS)         .SVC Ø54, DISABLE
DC    A(AROUND_PROCESSORS)         .SVC Ø55, EOV
DC    A(AROUND_PROCESSORS)         .SVC Ø56, ENQ/RESERVE
DC    A(AROUND_PROCESSORS)         .SVC Ø57, FREEDBUF
DC    A(AROUND_PROCESSORS)         .SVC Ø58, RELBUF/REQBUF
DC    A(AROUND_PROCESSORS)         .SVC Ø59, OLTEP
DC    A(AROUND_PROCESSORS)         .SVC Ø6Ø, (E)STAE
DC    A(AROUND_PROCESSORS)         .SVC Ø61, IKJEG6A
DC    A(AROUND_PROCESSORS)         .SVC Ø62, DETACH
DC    A(AROUND_PROCESSORS)         .SVC Ø63, CHKPT
DC    A(INFO_RDJFCB)               .SVC Ø64, RDJFCB
DC    A(AROUND_PROCESSORS)         .SVC Ø65, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø66, BTAMTEST
DC    A(AROUND_PROCESSORS)         .SVC Ø67, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø68, SYNADAF/RLS
DC    A(AROUND_PROCESSORS)         .SVC Ø69, BSP
DC    A(AROUND_PROCESSORS)         .SVC Ø7Ø, GSERV
DC    A(AROUND_PROCESSORS)         .SVC Ø71, ASGNBFR/....
DC    A(AROUND_PROCESSORS)         .SVC Ø72, ?????????
DC    A(AROUND_PROCESSORS)         .SVC Ø73, SPAR
DC    A(AROUND_PROCESSORS)         .SVC Ø74, DAR
DC    A(AROUND_PROCESSORS)         .SVC Ø75, DQUEUE
DC    A(AROUND_PROCESSORS)         .SVC Ø76, ????????
DC    A(AROUND_PROCESSORS)         .SVC Ø77, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø78, LSPACE
DC    A(AROUND_PROCESSORS)         .SVC Ø79, STATUS
DC    A(AROUND_PROCESSORS)         .SVC Ø8Ø, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø81, SETPRT/DEV
DC    A(AROUND_PROCESSORS)         .SVC Ø82, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø83, SMF(E)WTM
DC    A(AROUND_PROCESSORS)         .SVC Ø84, GRAPHICS
DC    A(AROUND_PROCESSORS)         .SVC Ø85, DDRSWAP
DC    A(AROUND_PROCESSORS)         .SVC Ø86, ATLAS
DC    A(AROUND_PROCESSORS)         .SVC Ø87, DOM
DC    A(AROUND_PROCESSORS)         .SVC Ø88, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø89, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø9Ø, RSVD
DC    A(AROUND_PROCESSORS)         .SVC Ø91, VOLSTAT
DC    A(AROUND_PROCESSORS)         .SVC Ø92, TCBEXCP
```

```
DC    A(AROUND_PROCESSORS)         .SVC Ø93, TGET/TPG/TPUT
DC    A(AROUND_PROCESSORS)         .SVC Ø94, STCC
DC    A(AROUND_PROCESSORS)         .SVC Ø95, SYSEVENT
DC    A(AROUND_PROCESSORS)         .SVC Ø96, STAX
DC    A(AROUND_PROCESSORS)         .SVC Ø97, IKJEGS9G
DC    A(AROUND_PROCESSORS)         .SVC Ø98, PROTECT
DC    A(AROUND_PROCESSORS)         .SVC Ø99, DYNALLOC
DC    A(AROUND_PROCESSORS)         .SVC 1ØØ, IKJEFFIB
DC    A(AROUND_PROCESSORS)         .SVC 1Ø1, QTIP
DC    A(AROUND_PROCESSORS)         .SVC 1Ø2, AQCTL
DC    A(AROUND_PROCESSORS)         .SVC 1Ø3, XLATE
DC    A(AROUND_PROCESSORS)         .SVC 1Ø4, TOPCTL
DC    A(AROUND_PROCESSORS)         .SVC 1Ø5, IMGLIB
DC    A(AROUND_PROCESSORS)         .SVC 1Ø6, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 1Ø7, MODESET
DC    A(AROUND_PROCESSORS)         .SVC 1Ø8, RSVD
DC    A(INFO_SVC1Ø9)               .SVC 1Ø9, ESR TYPE 4
DC    A(AROUND_PROCESSORS)         .SVC 11Ø, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 111, ?????????
DC    A(AROUND_PROCESSORS)         .SVC 112, PGRLSE
DC    A(AROUND_PROCESSORS)         .SVC 113, PGFIX/FREE/LOAD..
DC    A(AROUND_PROCESSORS)         .SVC 114, EXCPVR
DC    A(AROUND_PROCESSORS)         .SVC 115, RSVD
DC    A(INFO_SVC116)               .SVC 116, ESR TYPE 1
DC    A(AROUND_PROCESSORS)         .SVC 117, DEBCHK
DC    A(AROUND_PROCESSORS)         .SVC 118, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 119, TESTAUTH
DC    A(INFO_SVC12Ø)               .SVC 12Ø, GETMAIN/FREEMAIN
DC    A(AROUND_PROCESSORS)         .SVC 121, VSAM
DC    A(INFO_SVC122)               .SVC 122, ESR TYPE 2
DC    A(AROUND_PROCESSORS)         .SVC 123, PURGEDQ
DC    A(AROUND_PROCESSORS)         .SVC 124, TPIO
DC    A(AROUND_PROCESSORS)         .SVC 125, EVENTS
DC    A(AROUND_PROCESSORS)         .SVC 126, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 127, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 128, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 129, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 13Ø, RACHECK
DC    A(AROUND_PROCESSORS)         .SVC 131, RACINIT
DC    A(AROUND_PROCESSORS)         .SVC 132, RACLIST
DC    A(AROUND_PROCESSORS)         .SVC 133, RACDEF
DC    A(AROUND_PROCESSORS)         .SVC 134, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 135, RSVD
DC    A(AROUND_PROCESSORS)         .SVC 136, RSVD
DC    A(INFO_SVC137)               .SVC 137, ESR TYPE 6
DC    A(AROUND_PROCESSORS)         .SVC 138, PGSER
DC    A(AROUND_PROCESSORS)         .SVC 139, CVAF
DC    A(AROUND_PROCESSORS)         .SVC 14Ø, ????
DC    A(AROUND_PROCESSORS)         .SVC 141, ????
DC    A(AROUND_PROCESSORS)         .SVC 142, ????
DC    A(AROUND_PROCESSORS)         .SVC 143, GENKEY/RETKEY..
DC    A(AROUND_PROCESSORS)         .SVC 144, OE PTRACE
```

```
         EJECT
INFO_WAIT      MODENTRY INITBASE=R1Ø,BAKR=YES
         MVC   FIELDS+15+2Ø(17),=CL17'COUNT=X''12345678'''
         UNPK  FIELDS+15+28(9),OLDREGS(5)
         MVI   FIELDS+15+28+8,C''''
         TR    FIELDS+15+28(8),HEXCHAR-C'Ø'
         IF    TM,OLDREGS+4,X'8Ø',Z
            MVC   FIELDS+15+4Ø(4),=C'ECB='
            UNPK  FIELDS+15+44(9),OLDREGS+4(5)
            MVI   FIELDS+15+52,X'4Ø'
            TR    FIELDS+15+44(8),HEXCHAR-C'Ø'
         ELSE
            L     R1,OLDREGS+4
            LCR   R1,R1
            ST    R1,DUB
            MVC   FIELDS+14+4Ø(8),=C'ECBLIST='
            UNPK  FIELDS+14+48(9),DUB(5)
            MVI   FIELDS+15+56,X'4Ø'
            TR    FIELDS+14+48(8),HEXCHAR-C'Ø'
         ENDIF
         MODEXIT
         EJECT
INFO_POST      MODENTRY INITBASE=R1Ø,BAKR=YES
         MVC   PRTLINE+6Ø(12),=CL12'ECB='
         MVC   PRTLINE+75(14),=CL14'CODE=X''ØØ1122'''
         UNPK  PRTLINE+82(7),OLDREGS+1(4)
         MVI   PRTLINE+88,C''''
         TR    PRTLINE+82(6),HEXCHAR-C'Ø'
         IF    TM,OLDREGS+4,X'8Ø',Z
            UNPK  PRTLINE+64(9),OLDREGS+4(5)
         ELSE
            L     R1,OLDREGS+4
            UNPK  PRTLINE+64(9),Ø(R1)
            MVC   PRTLINE+9Ø(13),=C'ASCB='
            UNPK  PRTLINE+95(9),4(5,R1)
            MVI   PRTLINE+1Ø3,X'4Ø'
            TR    PRTLINE+95(8),HEXCHAR-C'Ø'
            IF    ICM,R15,15,4(R1),NZ
               MVC   PRTLINE+1Ø5(12),=C'ASID=X''1234'''
               UNPK  PRTLINE+112(5),ASCBASID-ASCB(3,R15)
               MVI   PRTLINE+117,C''''
               TR    PRTLINE+112(4),HEXCHAR-C'Ø'
            ENDIF
            IF    TM,OLDREGS,X'8Ø',O
               MVC   PRTLINE+12Ø(8),=CL8'KEY='
               XR    R15,R15
               IC    R15,12(,R1)
               CVD   R15,DUB
               OI    DUB+7,X'ØF'
               UNPK  PRTLINE+124(2),DUB+6(2)
            ENDIF
         ENDIF
```

```
         MVI   PRTLINE+72,X'4Ø'
         TR    PRTLINE+64(8),HEXCHAR-C'Ø'
         MODEXIT
         EJECT
INFO_LINK_LOAD MODENTRY INITBASE=R1Ø,BAKR=Y .LINK/LOAD/DELETE SVC 6/8/9
         MVC   PRTLINE+65(11),=CL11'EP='
         IF    CLI,CODEFLD+1,EQ,6        .LINK?
            L     R15,OLDREGS+15*4       .R15=A(EP,DCB)
            LM    R1,R2,Ø(R15)
         ELSE
            LM    R1,R2,OLDREGS          .LOAD: RØ=A(EP),R1=A(DCB)
         ENDIF
         IF    LTR,R1,R1,M                  .IF POS, R1=A(EP)
            LCR   R1,R1                     .A(DE LIST IN 2'S COMP)
         ENDIF
         MVC   PRTLINE+68(8),Ø(R1)       .EPNAME
* FOR LOAD/LINK:
* IF DCB ADDR NOT ZERO, IT MUST BE VALID AND OPEN, ELSE THE SVC WOULD
* HAVE FALLEN OVER.
         IF    CLI,CODEFLD+1,NE,9,AND,N,R2,=A(X'7FFFFFFF'),NZ
            MVC   PRTLINE+8Ø(15),=CL15'DDNAME='
            L     R15,PSATOLD-PSA
            L     R15,TCBTIO-TCB(,R15)
            AH    R15,DCBTIOT-IHADCB(,R2)
            MVC   PRTLINE+87(8),TIOEDDNM-TIOENTRY(R15)
         ENDIF
         MODEXIT
         EJECT
INFO_SVC1Ø     MODENTRY INITBASE=R1Ø,BAKR=YES
         IF    TM,OLDREGS+4,X'8Ø',O
            MVC   PRTLINE+53(16),=CL16'GETMAIN'
         ELSE
            MVC   PRTLINE+53(16),=CL16'FREEMAIN'
         ENDIF
         XR    R1,R1
         IC    R1,OLDREGS
         CVD   R1,DUB
         OI    DUB+7,X'ØF'
         MVC   PRTLINE+65(6),=CL6'SP=ØØØ'
         UNPK  PRTLINE+68(3),DUB+6(2)
         IF    ICM,R1,B'Ø111',OLDREGS+1,NZ  .LEN Ø, SUBPOOL FREEMAIN
            MVC   PRTLINE+75(13),=C'LEN=X''ØØØØØØ'''
            UNPK  PRTLINE+81(7),OLDREGS+1(4)
            MVI   PRTLINE+87,C''''
            TR    PRTLINE+81(6),HEXCHAR-C'Ø'
            IF    TM,OLDREGS+4,X'8Ø',Z
               MVC   PRTLINE+9Ø(5),=C'ADDR='
               UNPK  PRTLINE+95(9),OLDREGS+4(5)
               MVI   PRTLINE+1Ø3,X'4Ø'
               TR    PRTLINE+95(8),HEXCHAR-C'Ø'
            ENDIF
         ENDIF
```

```
         MODEXIT
         EJECT
INFO_BLDL_FIND MODENTRY INITBASE=R1Ø,BAKR=YES
         IF    ICM,R1,15,OLDREGS+4,P
            MVC   PRTLINE+53(9),=CL9'BLDL'
         ELSE
            MVC   PRTLINE+53(9),=CL9'FIND'
            LPR   R1,R1
         ENDIF
         IF    LTR,R1,R1,NZ
            MVC   PRTLINE+6Ø(15),=CL15'DDNAME='
            L     R15,PSATOLD-PSA
            L     R15,TCBTIO-TCB(,R15)
            AH    R15,DCBTIOT-IHADCB(,R1)
            MVC   PRTLINE+67(8),TIOEDDNM-TIOENTRY(R15)
         ENDIF
         MODEXIT
         EJECT
INFO_OPEN_CLOSE    MODENTRY                                          +
               INITBASE=R1Ø,BAKR=YES
         IF    ICM,R1,15,OLDREGS+4,Z
            MVC   PRTLINE+6Ø(7),=C'MODE=31'
            L     R1,OLDREGS
         ELSE
            MVC   PRTLINE+6Ø(7),=C'MODE=24'
         ENDIF
         IF    TM,Ø(R1),X'8Ø',Z
            MVC   PRTLINE+7Ø(16),=C'DCB LIST, START='
            ST    R1,DUB
            UNPK  PRTLINE+86(9),DUB(5)
            TR    PRTLINE+86(8),HEXCHAR-C'Ø'
            MVI   PRTLINE+94,X'4Ø'
         ELSE
            IF    CLC,=F'Ø',EQ,OLDREGS+4
               L     R2,4(,R1)
            ELSE
               XR    R2,R2
               ICM   R2,7,1(R1)
            ENDIF
            MVC   PRTLINE+7Ø(15),=CL15'DDNAME='
            IF    TM,DCBOFLGS-IHADCB(R2),DCBOFOPN,Z
               MVC   PRTLINE+77(8),DCBDDNAM-IHADCB(R2)
            ELSE
               L     R15,PSATOLD-PSA
               L     R15,TCBTIO-TCB(,R15)
               AH    R15,DCBTIOT-IHADCB(,R2)
               MVC   PRTLINE+77(8),TIOEDDNM-TIOENTRY(R15)
            ENDIF
         ENDIF
         MODEXIT
         EJECT
INFO_STOW      MODENTRY                                             +
```

```
              INITBASE=R1Ø,BAKR=YES
         MVC   PRTLINE+61(5),=C'TYPE='
         MVC   PRTLINE+7Ø(7),=C'DDNAME='
         SELECT
         WHEN  ICM,R1,15,OLDREGS+4,M
            LCR   R1,R1
            IF    ICM,RØ,15,OLDREGS,M
               MVI   PRTLINE+66,C'C'
            ELSE
               MVI   PRTLINE+66,C'R'
            ENDIF
         WHEN  ICM,RØ,15,OLDREGS,M
            MVI   PRTLINE+66,C'D'
         WHEN  CC=8
            MVI   PRTLINE+66,C'I'
         WHEN  NONE
            MVI   PRTLINE+66,C'A'
         ENDSEL
         L     R15,PSATOLD-PSA
         L     R15,TCBTIO-TCB(,R15)
         AH    R15,DCBTIOT-IHADCB(,R1)
         MVC   PRTLINE+77(8),TIOEDDNM-TIOENTRY(R15)
         IF    LTR,R1,RØ,M
            LCR   R1,R1
         ENDIF
         SELECT
         WHEN  CLI,PRTLINE+66,EQ,C'A'
            MVC   PRTLINE+9Ø(7),=C'MEMBER='
            MVC   PRTLINE+97(8),Ø(R1)
         WHEN  CLI,PRTLINE+66,EQ,C'C'
            MVC   PRTLINE+9Ø(8),=C'OLDNAME='
            MVC   PRTLINE+98(8),Ø(R1)
            MVC   PRTLINE+11Ø(8),=C'NEWNAME='
            MVC   PRTLINE+118(8),8(R1)
         WHEN  CLI,PRTLINE+66,EQ,C'D'
            MVC   PRTLINE+9Ø(7),=C'MEMBER='
            MVC   PRTLINE+97(8),Ø(R1)
         WHEN  CLI,PRTLINE+66,EQ,C'R'
            MVC   PRTLINE+9Ø(7),=C'MEMBER='
            MVC   PRTLINE+97(8),Ø(R1)
         ENDSEL
         MODEXIT
         EJECT
INFO_ATTACH    MODENTRY                                              +
               INITBASE=R1Ø,BAKR=YES
         MVC   PRTLINE+65(3),=C'EP='
         L     R1,OLDREGS+15*4
         L     R15,Ø(,R1)
         MVC   PRTLINE+68(8),Ø(R15)
         IF    ICM,R1,15,4(R1),NZ
            MVC   PRTLINE+8Ø(7),=C'DDNAME='
            L     R15,PSATOLD-PSA
```

```
            L     R15,TCBTIO-TCB(,R15)
            AH    R15,DCBTIOT-IHADCB(,R1)
            MVC   PRTLINE+87(8),TIOEDDNM-TIOENTRY(R15)
         ENDIF
         MODEXIT
         EJECT
INFO_RDJFCB    MODENTRY                                              +
               INITBASE=R1Ø,BAKR=YES
         L     R1,OLDREGS+4
         IF    TM,Ø(R1),X'8Ø',NO
            MVC   PRTLINE+6Ø(16),=C'DCB LIST, START='
            UNPK  PRTLINE+76(9),OLDREGS+4(5)
            MVI   PRTLINE+85,X'4Ø'
            TR    PRTLINE+76(8),HEXCHAR-C'Ø'
         ELSE
            L     R1,Ø(,R1)
            MVC   PRTLINE+6Ø(7),=C'DDNAME='
            IF    TM,DCBOFLGS-IHADCB(R1),DCBOFOPN,Z
               MVC   PRTLINE+67(8),DCBDDNAM-IHADCB(R1)
            ELSE
               L     R15,PSATOLD-PSA
               L     R15,TCBTIO-TCB(,R15)
               AH    R15,DCBTIOT-IHADCB(,R1)
               MVC   PRTLINE+67(8),TIOEDDNM-TIOENTRY(R15)
            ENDIF
         ENDIF
         MODEXIT
         EJECT
INFO_SVC1Ø9    MODENTRY                                              +
               INITBASE=R1Ø,BAKR=YES
         L     R15,OLDREGS+(15*4)
         SELECT
         WHEN  C,R15,EQ,=F'5'
            MVC   PRTLINE+63(1Ø),=CL1Ø'-GTFSRV'
         WHEN  C,R15,EQ,=F'7'
            MVC   PRTLINE+63(1Ø),=CL1Ø'-MFSTART'
         WHEN  C,R15,EQ,=A(X'16')
            MVC   PRTLINE+63(12),=CL12'-MFDATA(RMF)'
         WHEN  C,R15,EQ,=A(X'18')
            MVC   PRTLINE+63(13),=CL13'-HSM,IGX00024'
         WHEN  C,R15,EQ,=A(X'1A'),OR,C,R15,EQ,=A(X'1B')
            MVC   PRTLINE+63(13),=CL13'-TSO/E'
         WHEN  C,R15,EQ,=A(X'1C')
            MVC   PRTLINE+63(13),=CL13'-ESPIE'
            SELECT
```

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

Pieter Wiid
Advisory Systems Engineer
Perestel (South Africa)

MVS news

Syncsort has begun shipping Release 3.7 of its SyncSort MVS sort and data manipulation tool for OS/390 systems. It has a new Parasort technique for parallel tape input, which is claimed to cut sort elapsed time. The product allows SyncSort to read input from two, three, or four tape drives simultaneously, and is said to improve input elapsed time performance up to 20% when two volumes are processed in parallel and up to 33% when four are processed. General algorithmic and optimization enhancements are claimed to have improved SyncSort performance over DFSORT Release 14.0, with TCB CPU time reduced by 35%, total CPU time by 44%, and EXCPs by 39%.

Gains in the processing of COBOL-invoked sorts have also been made. Features added to the existing year 2000 readiness include new Y2K-specific data formats and a user-defined century window, which allows the correct interpretation of two-digit years and conversion to four-digit years when required. The software can also be used to select production data for test cases, age date fields, and replace COBOL programs that select, summarize, reformat, and report on various types of data.

For further information contact:
Syncsort Inc, 50 Tice Blvd, Woodcliffe Lake, NJ 07675, USA.
Tel: (201) 930 8200
Fax: (201) 930 8290 or

Syncsort Ltd, 60 Churchill Square, Kings Hill, West Malling, Kent, ME19 4DU, UK.
Tel: (01732) 849000
Fax: (01732) 875215.

* * *

Vanguard Integrity Professionals has announced Version 2.1 of its Vanguard RioVision Windows interface for OS/390 Security Server administration, providing mainframe security administrators with a view of IBM Security Server (RACF) and DB2 security data.

The Security Server group tree is represented in an Internet Explorer-type graphical format and the administrator can add or remove permissions, modify group or user profiles, add new users, and perform other security tasks.

Among the new bits is context-sensitive help down to the field level, as well as balloon help for all tool bar buttons. Also, RioVision now shows existing 'managed' and 'peer' relationships, letting administrators define, approve, and un-define these relationships. It automatically generates the necessary RACLINK commands.

The new QUERY operator command displays the current RioVision/Security Server settings, the number and names of users currently signed onto RioVision, and the status of the maintenance that has been applied to the product.

Vanguard Integrity Professionals, Inc, 180 S. Anita Drive, Orange, CA 92668-3306, USA.
Tel: (714) 939 0377
Fax: (714) 939 0273.

* * *