April 1998

In this issue

3  Non-disruptive START command
18 EXEC CICS ADDRESS CSA revisited
21 CICS statement tool
31 Easing the transition into STGPROT=YES
38 CICS date simulator for year 2000 testing – part 2
48 CICS news

© Xephon plc 1998
CICS Update

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38030
From USA: 01144 1635 38030
E-mail: xephon@compuserve.com

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

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

Subscriptions and back-issues
A year’s subscription to CICS Update, comprising twelve monthly issues, costs £170.00 in the UK; $260.00 in the USA and Canada; £176.00 in Europe; £182.00 in Australasia and Japan; and £182.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1994 issue, are available separately to subscribers for £14.50 ($22.00) each including postage.

Editor
Robert Burgess

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

Contributions
Articles published in CICS Update are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our Notes for Contributors.

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

© Xephon plc 1998. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher. Copying permits are available from Xephon in the form of pressure-sensitive labels, for application to individual copies. A pack of 240 labels costs $36 (£24), giving a cost per copy of 15 cents (10 pence). To order, contact Xephon at any of the addresses above.

Printed in England.
Non-disruptive START command

This article, the fourth in the series, continues to examine some of the options and features of the API and SPI. A partial discussion of these commands and programs was presented at Xephon’s CICS Update conference held in London in December 1997.

The main topic of this article is how to implement a non-disruptive START command.

The source code language used to illustrate the concepts is COBOL written to ANSI 85 standards; the BMS macros provided can be converted to the SDF II (and probably other) screen ‘painting’ packages.

NON-DISRUPTIVE MESSAGE DELIVERY

The SPI is a very powerful area where sophisticated function can be introduced to an application to make it more ‘user-friendly’. An example of a ‘non-friendly’ application is CMSG, the IBM-supplied message-switching transaction of BMS. If this is used to deliver a message to a terminal, it will disrupt any pseudo-conversational transaction sequence that the user is running. In fact, any application that schedules a transaction at an interactive terminal runs the same risk.

To demonstrate how this problem can be overcome, I have written a sample program which implements a non-disruptive method of message delivery from one terminal to another. The application is hardly complete – it consists of two programs and a BMS mapset consisting of two maps. Using these, I will discuss the techniques required to allow you to restore the user’s environment if you interrupt what they are doing.

Note that both programs use SPI commands and so need to use the SP translator option, which is why the CBL XOPTS(SP) statement is included as the first line of both the programs.
MESSAGE ACCEPTANCE PROGRAM

The first program accepts the text of a message and the CICS terminal identifier to which the message is to be delivered. The program is fairly unremarkable, except that it uses what I call a ‘defensive programming’ technique. These are methods that are designed to catch out supposedly ‘impossible’ error conditions as soon as they arise. In this case, it tests EIBCALEN for a specific value rather than simply assuming that a non-zero value is valid.

The two programs are mostly independent of the names you choose to call them, as well as the transaction code names you select. However, since this program must know what transaction to START, it assumes that the names of the transactions are related to one another such that the first three characters of the two transactions are identical, the names varying only in the last position. The program is set up so that it expects the last character of the message delivery transaction to be ‘D’. If you wish to use something different change the last position of WS-DELIVERY-TRANSID.

The program also uses an INQUIRE TERMINAL command to verify that the identifier entered by the user is valid. The program and map could be modified to allow entry of a NETNAME as an alternative. Note that the program uses XXXXMAP as the name of both the map and the mapset – if you wish to change it, use a global change for that name.

The program uses a delay on the START command with the AFTER SECONDS(WS-DELAY-SECONDS) option. The program is set up to use a five-second delay, but you can change it to something more suitable. It can be in the range 0-359,999 when HOURS and MINUTES are not also specified (which is the way the START command is coded in the program as supplied).

A few other minor points to note about the message acceptance program:

- It uses the ‘@’ symbol as a 3270-attribute. This is the combination required for ASKIP, DRK which is one aggregate IBM has failed to include in DFHBMSCA.
- It replaces nulls in the input with spaces (since nulls are not transmitted).
• It eliminates any trailing blank lines of the input message.

MESSAGE DELIVERY PROGRAM

The other program, which performs the actual message delivery, is more interesting. It uses quite a variety of infrequently used commands and options and also actualizes the technique to ensure that, after reading the message, the user can go back to whatever (s)he was doing. The overall flow is as follows:

• Find out how the transaction was started.

  This is obtained via the STARTCODE option of the ASSIGN command. The program checks that it has been started in the proper way, to ensure that no unnecessary disruption is caused, which again is really a ‘defensive programming’ technique. This is done by checking that there is a DFHCOMMAREA if it has been initiated by terminal input.

  When the message delivery task is begun via a START command at a terminal, it needs to save the next transaction identifier and also any DFHCOMMAREA data stored by the previous pseudo-conversational transaction. It ensures that it is associated with a terminal by using the FACILITY option of the ASSIGN command.

  – Obtain the NEXTTRANSID using the INQUIRE TERMINAL command.
  – Prevent any other Automatic Task Initiation (ATI) whilst the message is being displayed using the NOATI option of the SET TERMINAL command.
  – Save the DFHCOMMAREA data saved by the previous pseudo-conversational task into a TS queue.
  – Ensure that there is a message to display. Logically there should always be a message for display; however, the general ‘defensive programming’ approach handles this logically-suspect situation with grace.

    If no FROM option was used on the START, the program frees up the keyboard via the FREEKB option of the SEND
CONTROL command and returns to CICS as appropriate.

– Secure the data currently displayed on the screen.

It uses a RECEIVE BUFFER command with the ASIS option to obtain it, ensuring no erroneous upper-case translation is performed and saving the cursor position as well.

– Get the passed data containing the message to be delivered. It does this via the standard RETRIEVE command.

– Send the message to the screen with a standard header.

• The message is delivered using a SEND MAP command, with a map name of ‘DELIVRY’ and a mapset name of ‘XXXXXMAP’. If either of these needs to be changed, use a global change for that name. Note that if you changed the mapset name for the message acceptance program, then it will need to be changed in this program as well.

– End the task but keep control in this application and save the previous screen data. This is done via a run-of-the-mill RETURN command.

• When the message delivery task is begun at a terminal (the user has read the message and pressed ENTER), it needs to restore the saved data from DFHCOMMAREA and re-establish the original environment.

– Redisplay the original screen. It performs this via SEND FROM and SEND CONTROL commands. The SEND CONTROL command was discussed in a previous article in this series, Little-known features of API and SPI, CICS Update, Issue 147, February 1998. Briefly, this can be used to specify options that the ordinary SEND FROM command cannot. These include values such as the position of the cursor (CURSOR), the sounding of the audible alarm (ALARM), the releasing of the keyboard (FREEKB), and the resetting of the modified data tags (FRSET).

– Re-enable ATI again using the ATI option of the SET TERMINAL command.
If there was no previously set transaction code, the program issues a simple RETURN command.

If there had been a previously set transaction code, then there may have been a COMMAREA. Therefore this must be obtained from TS, the queue deleted, and the task ended with a RETURN command including the RETURN and COMMAREA options. Note that the program also handles the situation where there may have been a previously set next transaction identifier, but no data saved for it.

Finally, note that there are three ‘illogical’ scenarios that the program might encounter. In all of these cases the program abends the transaction. You may wish to change the codes used in these situations. The area for these codes is called ABEND-CODES and contains the sub-fields AC-WIEIRD-START, AC-NO-TERMINAL, and AC-FATAL-ERROR.

ACCEPTANCE PROGRAM

CBL XOPTS(SP)
IDENTIFICATION DIVISION.
PROGRAM-ID. SAMPLE.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
COPY XXXXMAP.
COPY DFHBMSCA.
COPY DFHAID.

FILLER.

WS-DELAY-SECONDS PIC S9(8) COMP VALUE 5.
WS-INDEX PIC S9(8) COMP.
WS-MSGLTH PIC S9(4) COMP.
WS-DERIVERY-TRANSID.
WS-DT-PREFIX PIC X(WS-INDEX).
FILLER PIC X(WS-INDEX) VALUE 'D'.
WS-IND1 PIC X(WS-INDEX) VALUE 'N'.
ERROR-FOUND VALUE 'Y'.
WS-IND2 PIC X(WS-INDEX) VALUE 'N'.
AT-LEAST-ONE-LINE VALUE 'Y'.
WS-IND3 PIC X(WS-INDEX) VALUE 'N'.
FOUND-END VALUE 'Y'.
FATAL-MSG PIC X(24) VALUE 'FATAL ERROR ENCOUNTERED!'.

PROCEDURE DIVISION.

IF EIBCALEN = 0
  * When first time, prompt the user for input.
  *
  PERFORM SEND-INITIAL
  MOVE LOW-VALUES TO WS-COMMAREA
  MOVE EIBTRMID TO WC-FROM
  EXEC CICS INQUIRE
    TERMINAL(EIBTRMID)
    USERID(WC-USER)
  END-EXEC
  PERFORM RET-CA
ELSE
  IF EIBCALEN = LENGTH OF DFHCOMMAREA
    MOVE DFHCOMMAREA TO WS-COMMAREA
  ELSE
    PERFORM FATAL-ERROR
  END-IF
END-IF

* Take the appropriate action requested by the user.
*

EVALUATE EIBAID
  WHEN DFHENTER
    PERFORM RECEIVE-IT
  WHEN DFHPF3
    PERFORM GET-OUT
WHEN DFHCLEAR
   PERFORM CLEAR-KEY
WHEN OTHER
   PERFORM INVALID-KEY
END-EVALUATE
.
RECEIVE-IT.
EXEC CICS RECEIVE
   MAP('XXXXMAP')
   NOHANDLE
END-EXEC
EVALUATE EIBRESP
WHEN DFHRESP(NORMAL)
   PERFORM MERGE-RTN
   MOVE LOW-VALUES TO XXXXMAPO
   MOVE '@' TO MSGA
   PERFORM VALIDATE-RTN
   IF ERROR-FOUND
      MOVE DFHBMASB TO MSGA
      IF AT-LEAST-ONE-LINE
         MOVE 'Ensure destination terminal is valid.'
         TO MSGO
      ELSE
         MOVE 'Message must have at least one line.'
         TO MSGO
      END-IF
   PERFORM SEND-DATAONLY
   PERFORM RET-CA
END-IF
MOVE DFHBMASB TO MSGA
MOVE WC-TERM TO AM-TERM
MOVE ACCEPTED-MSG TO MSGO
PERFORM VARYING WS-INDEX FROM 1 BY 1
   UNTIL WS-INDEX > 948
      IF WC-CHAR(WS-INDEX) = LOW-VALUE
         MOVE SPACE TO WC-CHAR(WS-INDEX)
      END-IF
END-PERFORM
MOVE LENGTH OF WC-DATA TO WS-MSGLTH
PERFORM VARYING WS-INDEX FROM 12 BY -1
   UNTIL FOUND-END
      IF WC-LINE(WS-INDEX) = SPACES
         SUBTRACT 79 FROM WS-MSGLTH
      ELSE
         SET FOUND-END TO TRUE
      END-IF
END-PERFORM
MOVE EIBTRNID TO WS-DT-PREFIX
EXEC CICS START
   TRANSID(WS-DELIVERY-TRANSID)
TERMID(WC-TERM)
FROM(WC-DATA)
LENGTH(WS-MSGLTH)
AFTER SECONDS(WS-DELAY-SECONDS)
END-EXEC
PERFORM SEND-MERGE
MOVE LOW-VALUES TO WC-MSGAREA
PERFORM RET-CA
WHEN DFHRESP(MAPFAIL)
  MOVE DFHBMASB TO MSGA
  MOVE 'Please enter data.' TO MSGO
  PERFORM SEND-EXISTING-CURSOR
  PERFORM RET-CA
WHEN OTHER
  PERFORM FATAL-ERROR
END-EVALUATE
.
MERGE-RTN.
IF (TERML NOT = ZERO)
  OR  TERMF = DFHBMEOF
  MOVE TERMI TO WC-TERM
END-IF
PERFORM VARYING WS-INDEX FROM 1 BY 1 UNTIL WS-INDEX > 12
  IF (LINEL(WS-INDEX) NOT = ZERO)
    OR  LINEF(WS-INDEX) = DFHBMEOF
    MOVE LINEI(WS-INDEX) TO WC-LINE(WS-INDEX)
  END-IF
END-PERFORM
.
VALIDATE-RTN.
EXEC CICS INQUIRE
  TERMINAL(WC-TERM)
  NOHANDLE
END-EXEC
IF  EIBRESP NOT = DFHRESP(NORMAL)
  SET ERROR-FOUND TO TRUE
  MOVE -1         TO TERML
  MOVE DFHBMBRY TO TERMA
END-IF
PERFORM VARYING WS-INDEX FROM 1 BY 1 UNTIL WS-INDEX > 12
  IF (WC-LINE(WS-INDEX) NOT = LOW-VALUES)
    AND (WC-LINE(WS-INDEX) NOT = SPACES    )
    SET AT-LEAST-ONE-LINE TO TRUE
  END-IF
END-PERFORM
IF  NOT AT-LEAST-ONE-LINE
  SET ERROR-FOUND TO TRUE
  MOVE -1         TO LINEL(1)
ENDOR  
.
INVALID-KEY.
MOVE 'Invalid key pressed.' TO MSGO
MOVE DFHBMASB TO MSGA
PERFORM SEND-EXISTING-CURSOR
PERFORM RET-CA
.
SEND-EXISTING-CURSOR.
EXEC CICS SEND
MAP('XXXXMAP')
CURSOR(EIBCP0SN)
DATAONLY
FREEKB
END-EXEC
.
CLEAR-KEY.
MOVE 'Invalid key pressed.' TO MSGO
MOVE DFHBMASB TO MSGA
PERFORM SEND-EXISTING-DATA
.
SEND-EXISTING-DATA.
MOVE WC-TERM TO TERMO
PERFORM VARYING WS-INDEX FROM 1 BY 1 UNTIL WS-INDEX > 12
  MOVE WC-LINE(WS-INDEX) TO LINEO(WS-INDEX)
END-PERFORM
PERFORM SEND-MERGE
PERFORM RET-CA
.
GET-OUT.
EXEC CICS SEND
FROM(END-MSG)
ERASE
END-EXEC
PERFORM RET
.
SEND-DATAONLY.
EXEC CICS SEND
MAP('XXXXMAP')
CURSOR
DATAONLY
FREEKB
FRSET
END-EXEC
.
SEND-MERGE.
EXEC CICS SEND
MAP('XXXXMAP')
ERASE
END-EXEC
SEND-INITIAL.
   EXEC CICS SEND
      MAP('XXXXMAP')
      MAPONLY
      ERASE
   END-EXEC

RET-CA.
   EXEC CICS RETURN
      TRANSID(EIBTRNID)
      COMMAREA(WS-COMMAREA)
   END-EXEC

RET.
   EXEC CICS RETURN
   END-EXEC

FATAL-ERROR.
   EXEC CICS SEND
      FROM(FATAL-MSG)
      ERASE
   END-EXEC
   PERFORM RET

DELIVERY PROGRAM

CBL XOPTS(SP)
IDENTIFICATION DIVISION.
PROGRAM-ID. SAMPLE.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
   01 ABEND-CODES.
      03 AC-WIERD-START PIC X(04) VALUE 'STNS'.
      03 AC-NO-TERMINAL PIC X(04) VALUE 'STNT'.
      03 AC-FATAL-ERROR PIC X(04) VALUE 'STFE'.

   01 WS-COMMAREA.
      03 WS-CPOSN PIC S9(4) COMP.
      03 WS-BUFLTH PIC S9(4) COMP.
      03 WS-TRANID PIC X(04).
      03 WS-INDICATOR PIC X(01).
         08 HAD-COMMAREA VALUE 'Y'.
         08 NO-COMMAREA VALUE 'N'.
      03 WS-BUFFER PIC X(2101).
EXEC CICS ASSIGN
    STARTCODE(WS-STARTCODE)
END-EXEC
IF TERMINAL-INPUT
    IF EIBCALEN = Ø
        EXEC CICS SEND
            FROM(WRONG-MSG)
            ERASE
    END-EXEC
    PERFORM RET
ELSE
    MOVE DFHCOMMAREA TO WS-COMMAREA
    EXEC CICS SEND
        FROM(WS-BUFFER)
        LENGTH(WS-BUFLTH)
        ERASE
    END-EXEC
    EXEC CICS SEND CONTROL
        CURSOR(WS-CPOSN)
        VALUE 'T'
    END-EXEC
    IF STARTED-TASK = 'S'
    ELSE
        MOVE DFHCOMMAREA TO WS-COMMAREA
        EXEC CICS SEND
            FROM(WS-BUFFER)
            LENGTH(WS-BUFLTH)
            ERASE
        END-EXEC
        EXEC CICS SEND CONTROL
            CURSOR(WS-CPOSN)
EXEC CICS SET
   TERMINAL(EIBTRMID)
   ATI
   NOHANDLE
END-EXEC
IF  WS-TRANID = SPACES
   PERFORM RET
ELSE
   IF  HAD-COMMAREA
      PERFORM MAKE-TSN
      EXEC CICS READQ TS
         QUEUE(TSQ-NAME)
         SET(ADDRESS OF LS-TSREC)
         LENGTH(TSQ-LEN)
         NOHANDLE
      END-EXEC
      IF  EIBRESP NOT = DFHRESP(NORMAL)
         PERFORM FATAL-ABEND
      END-IF
      PERFORM DELETE-TSQ
      EXEC CICS RETURN
         TRANSID(WS-TRANID)
         COMMAREA(LS-TSREC)
         LENGTH(TSQ-LEN)
      END-EXEC
   ELSE
      PERFORM RET-TRANSID
   END-IF
END-IF
ELSE
   IF  NOT STARTED-TASK
      EXEC CICS ABEND
         ABCODE(AC-WEIRD-START)
   END-EXEC
END-IF
EXEC CICS ASSIGN
   FACILITY(WS-TERM)
   NOHANDLE
END-EXEC
IF  EIBRESP = DFHRESP(INVREQ)
   EXEC CICS ABEND
      ABCODE(AC-NO-TERMINAL)
END-EXEC
EXEC CICS INQUIRE
   TERMINAL(EIBTRMID)
   NEXTTRANSID(WS-TRANID)
   NOHANDLE
EXEC CICS SET
   TERMINAL(EIBTRMID)
   NOATI
   NOHANDLE

END-EXEC
IF EIBCALEN > Ø
   MOVE EIBCALEN TO TSQ-LEN
   PERFORM MAKE-TSN
   PERFORM DELETE-TSQ
   EXEC CICS WRITEQ TS
      QUEUE(TSQ-NAME)
      FROM(DFHCOMMAREA)
      LENGTH(TSQ-LEN)

END-EXEC
SET HAD-COMMAREA TO TRUE
ELSE
   SET NO-COMMAREA TO TRUE
END-IF
IF NOT DATA-PROVIDED
   EXEC CICS SEND CONTROL
      FREEKB
END-EXEC
IF WS-TRANID = SPACES
   PERFORM RET
ELSE
   IF HAD-COMMAREA
      PERFORM DELETE-TSQ
      EXEC CICS RETURN
         TRANSID(WS-TRANID)
         COMMAREA(DFHCOMMAREA)
         LENGTH(TSQ-LEN)
   END-EXEC
   ELSE
      PERFORM RET-TRANSID
   END-IF
END-IF
MOVE LENGTH OF WS-BUFFER TO WS-BUFLTH
EXEC CICS RECEIVE BUFFER
   ASIS INTO(WS-BUFFER)
   LENGTH(WS-BUFLTH)
   NOHANDLE

END-EXEC
MOVE EIBCPOSN TO WS-CPOSN
MOVE LENGTH OF WS-INPUT TO WS-MSGLTH
EXEC CICS RETRIEVE
   INTO(WS-INPUT)
   LENGTH(WS-MSGLTH)
   NOHANDLE
END-EXEC
MOVE LOW-VALUES TO DELIVRYO
MOVE WS-FROM TO TERMIDO
MOVE WS-USER TO USERIDO
COMPUTE WS-IX = (WS-MGLTH - 12) / LENGTH OF WS-LINE
PERFORM UNTIL WS-IX = Ø
   MOVE WS-LINE(WS-IX) TO THEMMSGO(WS-IX)
   SUBTRACT 1 FROM WS-IX
END-PERFORM
EXEC CICS SEND
  MAP('DELIVRY')
  MAPSET('XXXXMAP')
  ERASE
END-EXEC
EXEC CICS RETURN
TRANSID(EIBTRNID)
COMMAREA(WS-COMMAREA)
END-EXEC
END-IF
.
MAKE-TSN.
MOVE EIBTRNID TO TSN-TRAN
MOVE EIBTRMID TO TSN-TERM
.
DELETE-TSQ.
EXEC CICS DELETEQ TS
  QUEUE(TSQ-NAME)
  NOHANDLE
END-EXEC
.
RET-TRANSID.
EXEC CICS RETURN
  TRANSID(WS-TRANID)
END-EXEC
.
RET.
EXEC CICS RETURN
END-EXEC
.
FATAL-ABEND.
EXEC CICS ABEND
  ABCODE(AC-FATAL-ERROR)
END-EXEC
.

BMS MACROS
* Acceptance screen
XXXXMAP  DFHMSD TYPE=SYSPARM,LANG=COBOL,MODE=INOUT,STORAGE=AUTO,
The final article in this series will continue the theme of using some of the useful but uncommonly used options and features of the API and SPI.

Jerry Ozaniec  
Circle Computer Group (UK)  
© Xephon 1998
EXEC CICS ADDRESS CSA revisited

INTRODUCTION

In *CICS/CA-IDMS 10.2 programs under CICS/ESA 4.1, CICS Update*, Issue 132, November 1996, I described how to modify CICS/CA-IDMS 10.2 load modules, allowing them to run under CICS/ESA 4.1. There was, in fact, another obstacle to overcome for the migration to be a success. The ‘EXEC CICS ADDRESS CSA’ command was used not only in the CICS/CA-IDMS 10.2 stub, but was actually pervasive in most of the client’s older applications. Unlike the dilemma presented by the CICS/CA-IDMS 10.2 stub, where the problem code was isolated to a small section, locating the illegal object code and superzapping them was not a viable solution. However, a solution had to be found – and soon.

PROBLEM DESCRIPTION

The original article described the incompatibility of CICS/CA-IDMS 10.2 and CICS/ESA 4.1. The CICS/CA-IDMS 10.2 stub used the ‘EXEC CICS ADDRESS CSA’ command and assumed that the Common Work Area (CWA) was exactly 512 bytes past the CSA. CICS/ESA 4.1 no longer supported the ‘EXEC CICS ADDRESS CSA’ command and the CWA was no longer contiguous with the CSA. I was able to solve the problem by developing superzap data that would convert the illegal instructions into valid ones. The primary reasons for this approach were that the client had low confidence in the currency of their application source code, and there was a pressing need to migrate to CA-IDMS 12.0.

As with many older CICS applications, programmers had often developed, or found a need to use, the ‘EXEC CICS ADDRESS CSA’ command – this being especially true for this particular client. The client enforced a standard compilation process, which automatically included common code that created date and time values from the CSA. Unfortunately, the same strictness was not diligently applied to their source code management.
In addition, there were no standards preventing the programmer from using the ‘EXEC CICS ADDRESS CSA’ command. Since CICS/ESA 4.1 does not support this command, it would seem that most customers would be forced to modify and recompile their programs. The prospect of recompiling questionable source code, and risking additional expense and time correcting the programs, was daunting.

WHAT TO DO?

There are packaged solutions to this problem, some of which capture the illegal command in the CICS Command Interface Exit (XEININ and XEIOUOUT), obtain the address of the CSA, and present the caller with the desired result. At the time, this seemed like a reasonable solution. Since I developed a solution for the CICS/CA-IDMS 10.2 problem, I felt that I could create one using the same approach as some of the vendor products. However, during my research I discovered another, far easier approach to this problem.

‘EXEC CICS ADDRESS CSA’ IS NOT SUPPORTED, UNLESS...

The CICS module DFHEEI is the EXEC interface processor for the DFHEIP ADDRESS, ASSIGN, PUSH, POP, and HANDLE commands. This seemed like a good place to start to see how I should design my exit. Fortunately, DFHEEI is still viewable via the View Program Listing (VPL) facility on IBMLink. As I scanned the source code, I noticed the following:

```
CLC 5(8,R10),=CL8'ADMASLC' GDDM CICS STUB?
BNE EIAR10 NO - ADDR PROTECTED CSA
```

In effect, what this code segment shows is that it honours the ‘EXEC CICS ADDRESS CSA’ request if you are IBM’s GDDM product. Otherwise, CICS would set the return value with an address that points to a fetch-protected storage area, thereby causing a program exception once the address is used. When I issued an ETR to IBM regarding the contents of the fetch-protected area, they stated that the information is proprietary. Apparently, IBM’s GDDM component that runs under CICS was not rewritten in time to be compliant with CICS/ESA 4.1. This exception provided me with the ‘back-door’ that I needed. As a result, I wrote a quick one-byte zap that converted the
BNE instruction shown above into an NOP instruction and, as hoped, a program compiled under CICS/VS 2.1.1 using the illegal instruction now worked. I later created an SMP/E usermod for the zap, to ensure that the modification would not disappear if maintenance were applied in the future. The usermod is written as follows:

```
++USERMOD(yourmodname)
++VER(C150) FMID(HCI4100) PRE(UN86657)
++ZAP(DFHEEI) DISTLIB(ADFHMOD)
   NAME DFHEEI
   VER 03C2 D507,A005,9090
   VER 03C8 4770,9098
   REP 03C8 4700,0000
```

I have not researched the feasibility of applying this modification to a CICS/ESA Version 3.3 system, but I suspect it can be done.

CONCLUSION

Under normal circumstances, I would not support this type of modification to system software. This practice typically causes problems for the customer when a major upgrade is planned and the customer has become overly dependent on their system modifications. However, I also believe that there are situations that make it very difficult for customers to migrate to a current level of software, especially if the functionality they need is dependent upon the software level of other related products, eg IDMS V12.0 requires CICS/ESA Version 3.3 or higher. In addition, forcing users to spend time (and money) modifying and recompiling programs that have been running for years without any problems is harsh, especially if the accuracy of the source code is questionable at best.

This modification provides users with some breathing space, allowing them to run their older applications on a higher level of CICS, and without having to support multiple versions of CICS.

Richard Tsujimoto
Consultant (USA)
CICS statement tool

INTRODUCTION
Because it is difficult to remember every CICS API statement, we have developed a menu-driven tool to help our programmers. This tool will help you to import the most common CICS-statements into a source program.

You can add the CICS-statement when you enter an ‘a’ (after) or ‘b’ (before) on your TSO/ISPF edit screen in the source code and call the REXX EXEC CICSAPI.

More statements can easily be added to the control file if required:

Command ===> cicsapi

<table>
<thead>
<tr>
<th>Action</th>
<th>VSAM HANDLING</th>
<th>QUEUE HANDLING</th>
</tr>
</thead>
<tbody>
<tr>
<td>-RF</td>
<td>(read file)</td>
<td>-ROTD (readqueue td)</td>
</tr>
<tr>
<td>-WF</td>
<td>(write file)</td>
<td>-ROTS (readqueue ts)</td>
</tr>
<tr>
<td>-RM</td>
<td>(rewrite file)</td>
<td>-WOTD (writequeue td)</td>
</tr>
<tr>
<td>-UL</td>
<td>(unlock file)</td>
<td>-WOTS (writequeue ts)</td>
</tr>
<tr>
<td>-D</td>
<td>(delete file)</td>
<td>-DQTD (deletequeue td)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-DQTS (deletequeue ts)</td>
</tr>
</tbody>
</table>

VSAM BROWSE HANDLING
- SB (start browse)
- RN (readnext)   MAP HANDLING
- RP (readprev)   - RM (receive map)
- EB (end browse) - SM (send map)
- RB (reset browse)

Figure 1: CICS statements generator
After this the menu-driven CICS statements generator will appear, as shown in Figure 1.

Suppose WF is entered, the following statements are inserted in the source program:

```
Command ===>  
****** *********************************** Top of Data ***
Ø001ØØ     THIS IS YOUR SOURCE CODE
Ø002ØØ  
=NOTE=     EXEC CICS WRITE
=NOTE=     FILE(FIlenAME)  
=NOTE=     MASSINSERT
=NOTE=     FROM(DATA-AREA)
=NOTE=     LENGTH(DATA-VALUE)
=NOTE=     RIDFLD(DATA-AREA)
=NOTE=     KEyLENGTH(DATA-VALUE)
=NOTE=     SYSSID(SYSTEMNAME)
=NOTE=     RBA|RRN  
=NOTE=     END-EXEC
****** **************************** Bottom of Data *
```

You can add these notes with ‘MMD....MMD’ (as line block commands). If you know the abbreviations, these can be entered directly in the first screen.

CICSAPI EXEC

```
/*REXX*/

/*Used panels / EXECs*/
/* - CICSAPI EXEC this REXX */
/* - CNTL the CICSAPI-database*/
/* - C@GEENAB: message panel */
/* - CICS1: panel 1*/
/* - CICS2: panel 2*/
/* - CICS3: panel 3*/
/* - CICS4: panel 4*/
/* - CICS5: panel 5*/
/*with 'md' you can make the statements permanent in the source*/
ADDRESS ISPEXEC  "LIBDEF ISPPLIB DATASET ID('YOUR.PANEL.LIBRARY')"

TRACE o
ISPEXEC 'Control Errors Return'
ISREDIT 'Macro (ACTIE) NOPROCESS'
UPPER ACTIE
/* if no a or b is placed as a line command */
```
ISREDIT 'PROCESS DEST'
if RC = 0 Then
  Do
    ISPEXEC 'ADDPOP ROW(2) COLUMN(10)'
    ISPEXEC 'DISPLAY PANEL(C@eenab)'
    exit
  end
ISREDIT '(ZDEST) = LINENUM .ZDEST'
/* Show first panel */
If ACTIE = '' Then
  Do
    ISPEXEC 'ADDPOP ROW(2) COLUMN(10)'
    ISPEXEC 'DISPLAY PANEL(CICS1)'
    RETC = RC
  End
/* if N pushed: show second panel */
If ACTIE = 'N' Then
  Do
    ISPEXEC 'ADDPOP ROW(2) COLUMN(10)'
    ISPEXEC 'DISPLAY PANEL(CICS2)'
    RETC = RC
  End
/* if N pushed: show third panel */
If ACTIE = 'N' Then
  Do
    ISPEXEC 'ADDPOP ROW(2) COLUMN(10)'
    ISPEXEC 'DISPLAY PANEL(CICS3)'
    RETC = RC
  End
/* if N pushed: show fourth panel */
If ACTIE = 'N' Then
  Do
    ISPEXEC 'ADDPOP ROW(2) COLUMN(10)'
    ISPEXEC 'DISPLAY PANEL(CICS4)'
    RETC = RC
  End
/* if N pushed: show fifth panel */
If ACTIE = 'N' Then Do
  Do
    ISPEXEC 'ADDPOP ROW(2) COLUMN(10)'
    ISPEXEC 'DISPLAY PANEL(CICS5)'
    RETC = RC
  End
End
/*if return code not equal 0, leave programm */
If RETC = 0 Then Exit
/*if variable actie is empty, call lees*/
If ACTIE = '' Then
  DO
    call lees
  END
End
/* read control file from database with matching CICS statement */
lees:
bestsds='YOUR.CONTROL.FILE(dataset)'
address tso
"allocate fi(indd1) da(''bestsds'') shr reuse"
k=1
"EXECIO * DISKR indd1 (STEM INDD1. FINIS)"
do i=1 to INDD1.Ø
    comm=substr(indd1.i,1,1Ø)
    comm=strip(comm)
    if comm=actie then
        do
            if k=1 then
                do
                    outdd1.k='       exec cics '||substr(indd1.i,2Ø,2Ø)
                end
            else
                do
                    OUTDD1.K='       '||substr(indd1.i,2Ø,7Ø)
                end
                k=k+1
            end
        end
    if k>1 then
        do
            OUTDD1.K='       end-exec'
            k=k+1
        end
    K=K-1
Do I=K BY -1 TO 1
    REGEL = outdd1.I
    IsrEdit "LINE_AFTER "ZDEST" = NOTELINE (REGEL)"
End
exit

MESSAGE PANEL

)BODY window(45 6)
+ %You didn't use after or before %
+ %        ===> PUSH F3 <=====% _Z
+ )INIT
    .ZVARS = ACTIE
)PROC
    VER(&ACTIE,NB,LIST)
)END
CICSAPI PANEL 1 (CICS1)

)BODY
+ % Generating CICS statements%
+
+ % These are the most used CICS statements%
+ % Push N for the next menu%
+
+Action %===>_Z +VSAM HANDLING QUEUE HANDLING
-RF (read file)   -RQTD (readqueue td)
-WF (write file)  -RQTS (readqueue ts)
-RW (rewrite file) -WQTD (writequeue td)
-UL (unlock file) -WQTS (writequeue ts)
-D (delete file)  -DQTD (deletequeue td)
-DQTS (deletequeue ts)

VSAM BROWSE HANDLING
-SB (start browse)
-RN (readnext) MAP HANDLING
-RP (readprev)   -RM (receive map)
-EB (end browse) -SM (send map)
-RB (reset browse)
+
)INIT
.ZVARS = ACTIE
)PROC
)END

CICSAPI PANEL 2 (CICS2)

)BODY
+ % Generating CICS statements%
+
+ % These are the least used CICS statements from A to D%
+ % Push N for the next menu%
+
+Action %===>_Z +CONTROL HANDLING
-FM (freemain)/GM (getmain)
-LI (link)
-XCTL (transfer program control)

-AB (abnormal end)
-AD (address)
-AS (assign)
-AT (asktime)/FT (formattime)
-CA (cancel)
-DEQ (dequeue)/ENQ (enqueue)
-DL (delay)
+
CICSAPI PANEL 3 (CICS3)

)BODY
+ % Generating CICS statements
+ % These are the least used CICS statements from H to R
+ % Push N for the next menu
+ +Action %=->_Z +HA (handle abnormal end)
   -HC (handle condition)
   -IC (ignore condition)
   -IP (inquire program)
   -IT (inquire transaction)
   -PO (post)
   -POH (pop handle)/PUH (push handle)
   -QS (query security)
   -REC (receive)
   -RL (release)
   -RTR (retrieve)
   -RTU (return)
+ )INIT
   .ZVARS = ACTIE
)PROC
)END

CICSAPI PANEL 4 (CICS4)

)BODY
+ % Generating CICS statements
+ % These are the least used CICS statements from S to Z
+ % Push N for the next menu
+ +Action %=->_Z +SC (spoolclose)/SR (spoolread)/SW (spoolwrite)
   -SU (suspend)
   -SENDT (send text)
   -SOF (signoff)/SON (signon)
   -ST (start)
   -SY (syncpoint)
   -VP (verify password)
+ )INIT
.ZVARS = ACTIE
)PROC
)END

CICSAPI PANEL 5 (CICS5)

)BODY
+ % Generating CICS statements%
+ % These are the CICS statements for DATACOMM.%
+ 
+Action -IABE (issue abend)
 %===>_Z +-ICON (issue connect)
 -IDISC (issue disconnect)
 -ICOPY (issue copy)
 -IERASEAUP (issue eraseaup)
 -IERR (issue error)
 -INOTE (issue note)
 -IPASS (issue pass)
 -IPREP (issue prepare)
 -IRESET (issue reset)
 -ISIGA (issue signal APPC)
 -CO (connect)
 -F (free)/FMRO (free mro)
 -RECA (receive APPC)/RECM (receive MRO)
 -RECD (receive display)
 -SENDA (send APPC)/SENDM (receive MRO)
 -SENDD (receive display)

+ )INIT
 .ZVARS = ACTIE
)PROC
)END

CICSAPI CONTROL STATEMENTS
This contains YOUR.CONTROL.FILE (dataset).

M_AB ABEND
M_AD ADDRESS
M_AS ASSIGN
M_AT ASKTIME
M_CA CANCEL
M_CO CONNECT
M_D DELETE
M_DEQ DEQ
M_DL DELAY
M_DQTD DELETEQ TD
M_DQTS DELETEQ TS

M_EB  ENDBR
M_ENQ  ENQ
M_F   FREE
M_FM  FREEMAIN
M_FMRO  FREE MRO
M_FT  FORMATTIME
M_GM  GETMAIN
M_HA  HANDLE ABEND
M_HC  HANDLE CONDITION
M_IABE  ISSUE ABEND
M_IABO  ISSUE ABORT
M_IADD  ISSUE ADD
M_ICON  ISSUE CONFIRMATION
M_ICOPY  ISSUE COPY
M_IDISC  ISSUE DISCONNECT
M_IEND  ISSUE END
M_IENDF  ISSUE ENDFILE
M_IENDOP  ISSUE ENDOUTPUT
M_IEODS  ISSUE EODS
M_IERASE  ISSUE ERASE
M_IERASEAUP  ISSUE ERASEAUP
M_IERR  ISSUE ERROR
M_ILOAD  ISSUE LOAD
M_INOTE  ISSUE NOTE
M_IPASS  ISSUE PASS
M_IPREP  ISSUE PREPARE
M_IPRINT  ISSUE PRINT
M_IQUERY  ISSUE QUERY
M_IREC  ISSUE RECEIVE
M_IREPL  ISSUE REPLACE
M_IRESET  ISSUE RESET
M_ISEND  ISSUE SEND
M_ISIGA  ISSUE SIGNAL
M_ISIGL  ISSUE SIGNAL
M_IWAIT  ISSUE WAIT
M_IC  IGNORE CONDITION
M_IP  INQUIRE PROGRAM
M_IT  INQUIRE TRANSACTION
M_LI  LINK
M_POH  POP HANDLE
M_PUH  PUSH HANDLE
M_PO  POST
M_QS  QUERY SECURITY
M_REC  RECEIVE
M_RECA  RECEIVE (APPC)
M_RECDO  RECEIVE (DISPLAY)
M_RECM  RECEIVE (MRO)
M_RF  READ FILE
M_RB  RESETBR
M_RL  RELEASE
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M_RM</td>
<td>RECEIVE MAP</td>
</tr>
<tr>
<td>M_RN</td>
<td>READNEXT</td>
</tr>
<tr>
<td>M_RP</td>
<td>READPREV</td>
</tr>
<tr>
<td>M_RQTD</td>
<td>READQ TD</td>
</tr>
<tr>
<td>M_RQTS</td>
<td>READQ TS</td>
</tr>
<tr>
<td>M_RTR</td>
<td>RETRIEVE</td>
</tr>
<tr>
<td>M_RTU</td>
<td>RETURN</td>
</tr>
<tr>
<td>M_SB</td>
<td>STARTBR</td>
</tr>
<tr>
<td>M_SC</td>
<td>SPOOLCLOSE</td>
</tr>
<tr>
<td>M_SENDA</td>
<td>SEND (APPC)</td>
</tr>
<tr>
<td>M_SENDDD</td>
<td>SEND (DISPLAY)</td>
</tr>
<tr>
<td>M_SENDM</td>
<td>SEND (MRO)</td>
</tr>
<tr>
<td>M_SENDT</td>
<td>SEND TEXT</td>
</tr>
<tr>
<td>M_SM</td>
<td>SEND MAP</td>
</tr>
<tr>
<td>M_SU</td>
<td>SUSPEND</td>
</tr>
<tr>
<td>M_SR</td>
<td>SPOOLREAD</td>
</tr>
<tr>
<td>M_SCW</td>
<td>SPOOLWRITE</td>
</tr>
<tr>
<td>M_SOF</td>
<td>SIGNOFF</td>
</tr>
<tr>
<td>M_SON</td>
<td>SIGNON</td>
</tr>
<tr>
<td>M_ST</td>
<td>START</td>
</tr>
<tr>
<td>M_SY</td>
<td>SYNCPOINT</td>
</tr>
<tr>
<td>M_UL</td>
<td>UNLOCK</td>
</tr>
<tr>
<td>M_VP</td>
<td>VERIFY PASSWORD</td>
</tr>
<tr>
<td>M_WF</td>
<td>WRITE FILE</td>
</tr>
<tr>
<td>M_WO</td>
<td>WRITE OPERATOR</td>
</tr>
<tr>
<td>M_WQTD</td>
<td>WRITEQ TD</td>
</tr>
<tr>
<td>M_WQTS</td>
<td>WRITEQ TS</td>
</tr>
<tr>
<td>M_XCTL</td>
<td>XCTL</td>
</tr>
<tr>
<td>AB</td>
<td>ABEND</td>
</tr>
<tr>
<td>AB</td>
<td>ABCODE(NAME)</td>
</tr>
<tr>
<td>AB</td>
<td>CANCEL</td>
</tr>
<tr>
<td>AB</td>
<td>NODUMP</td>
</tr>
<tr>
<td>AD</td>
<td>ADDRESS</td>
</tr>
<tr>
<td>AD</td>
<td>COMMAREA(PTR-REF)</td>
</tr>
<tr>
<td>AD</td>
<td>ACEE(PTR-REF)</td>
</tr>
<tr>
<td>AD</td>
<td>CWA(PTR-REF)</td>
</tr>
<tr>
<td>AD</td>
<td>EIB(PTR-REF)</td>
</tr>
<tr>
<td>AD</td>
<td>TCTUA(PTR-REF)</td>
</tr>
<tr>
<td>AD</td>
<td>TWA(PTR-REF)</td>
</tr>
<tr>
<td>AS</td>
<td>ASSIGN</td>
</tr>
<tr>
<td>AS</td>
<td>ABCODE(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>ABDUMP(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>ABPROGRAM(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>ALTSCRNHT(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>ALTSCRNWD(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>APLKYBD(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>APLTEXT(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>APPLID(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>ASRAINTRPT(DATA-AREA)</td>
</tr>
<tr>
<td>AS</td>
<td>ASRAKEY(CVDA)</td>
</tr>
</tbody>
</table>
Editor’s note: this article will be continued next month.

Paul Jansen (with special thanks to Marco Seesing & Martijn Bosschieter)
Systems Programmer
Interpay (The Netherlands) © M Bosschieter/ M Seesing 1998
Easing the transition into STGPROT=YES

In the past, when taking advantage of new CICS features, we have often found the need for tools to help ease the transition. We are currently implementing the STGPROT feature of CICS Version 4.1 (change the SIT parm from NO to YES, then cold start) and have found that some programs which previously worked normally in USER_KEY now abend with S0C4/SR0001 because they reference MVS or CICS internal areas.

If we turn on STGPROT=YES, the applications programmer’s progress may be hindered because systems programmers are frequently called to set programs to CICS_KEY. This makes life difficult for everybody. The operational concept is to enable STGPROT=YES, with the minimum impact to applications programmers, by automatically setting the program caught in an SR0001 abend to CICS_KEY, while informing the systems programmers that a program has been set to CICS_KEY to continue processing.

We use the XDUREQC global user exit to enable this operational concept. The skeleton XDUREQ exit DFH$XDRQ has been supplied by IBM in CICS410.SDFHSAMP, using the CICS XPI interface and MVS services (WTO) to do the following:

- GETMAIN required working storage.
- INQUIRE the execution key of the offending program.
- WTO to inform systems programmer of the SR0001 occurrence.
- Set the execution key of the offending program.
- WTO to inform systems programmer USER_KEY program has been set to CICS_KEY.
- FREEMAIN acquired working storage.

In addition, we code the following in a DFHPLTPI program:

- Suppress SR0001 SVC dumps:
  
  ```
  EXEC CICS SET SYSDUMPCODE('SR0001') NOSYSDUMP ADD
  ```

Enable XDUREQC exit:

EXEC CICS ENABLE PROGRAM('DFH$XDRQ') EXIT('XDUREQC') START

DFH$XDRQ

EXEC PGM=IEV9Ø,
EXEC REGION=4096K,
EXEC PARM='NODECK,OBJECT,XREF(SHORT)'

DD DSN=CICS.REL41.SDFHMAC,DISP=SHR
DD DSN=SYS1.MACLIB,DISP=SHR
DD DISP=SHR, DSN=SYS1.AMODGEN

DD UNIT=SYSALLDA,SPACE=(1700,(400,400))
DD UNIT=SYSALLDA,SPACE=(1700,(400,400))
DD UNIT=SYSALLDA,SPACE=(1700,(400,400))

DD DSN=&LOADSET,
UNIT=SYSALLDA,DISP=(,PASS),
SPACE=(400,(100,100,1))

DD SYSOUT=*
DD *

***********************************************************************
* MODULE NAME = DFH$XDRQ
* DESCRIPTIVE NAME = CICS/ESA ....
* SAMPLE USER EXIT PROGRAM FOR THE XDUREQC EXIT .
* MODIFIED BY CHORNG S. (JACK) HWANG
* HSA SYSTEM INC FOR STGPROT IMPLEMENTATION
* XDUREQC EXIT LOGIC:
* 1) SEE IF THIS IS SR0001 ABEND
* 2) VERIFY PROGRAM IS RDO/GRPLIST/AUTO INSTALL
* 3) WTO PROGRAM PROPERTIES
* 4) DELETE PROGRAM
* 5) INSTALL PROGRAM
* 6) WTO NEW PROGRAM INSTALL
* 5655-018
* COPYRIGHT = NONE
* STATUS = 4.1.0
* NOTES :
* DEPENDENCIES = S/370
* REQUIRES APAR PN61792
* RESTRICTIONS = NONE
* PATCH LABEL = NONE
* RESTRICTIONS = NONE
* PATCH LABEL = NONE
* PROCESSOR = ASSEMBLER
* ATTRIBUTES = READ ONLY, REENTRANT
*
* ENTRY POINT = DFH$XDRQ
*
* LINKAGE = INVOKED FROM THE XDUREQC USER EXIT CALL .
*
* INPUT = REGISTER 1 - ADDRESS OF DFHUEPAR
*
* DFHUEPAR CONTAINS THE FOLLOWING EXIT SPECIFIC PARAMETERS
*
* UEPTRANID ADDRESS OF THE 4-BYTE TRANSACTION-ID
*
* UEPUSER ADDRESS OF THE 8-BYTE USER-ID
*
* UEPTERM ADDRESS OF THE 4-BYTE TERMINAL-ID
*
* UEPPROG ADDRESS OF THE 8-BYTE APPLICATION PROGRAM NAME.
*
* UEPDUMPC ADDRESS OF COPY OF THE 8-BYTE DUMP CODE.
*
* UEPABCDE ADDRESS OF A COPY OF THE 8-BYTE KERNEL ERROR
*    CODE IN THE FORMAT XXX/YYYY.
*
* UEPDUMPT ADDRESS OF THE 1-BYTE DUMP TYPE. EITHER:
*    UEPDTRAN TRANSACTION DUMP REQUESTED
*    UEPDSYST SYSTEM DUMP REQUESTED
*
* THE FOLLOWING FIELDS REPRESENT THE DUMP TABLE ENTRY FOR THE
* DUMPCODE AT UEPDUMPC.
*
* UEPXDSCP ADDRESS OF A 1-BYTE DUMPSCOPE SETTING. EITHER:
*    UEPXDLOC DUMP ONLY ON LOCAL MVS IMAGE
*    UEPXDREL DUMPS TAKEN ON RELATED MVS IMAGES
*
* UEPXDTXN ADDRESS OF A 1-BYTE TRANDUMP SETTING. EITHER:
*    UEPXDYES A TRANSACTION DUMP WOULD BE TAKEN
*    UEPXNO A TRANSACTION DUMP WOULD NOT BE TAKEN
*
* UEPXDSYS ADDRESS OF A 1-BYTE SYSDUMP SETTING. EITHER:
*    UEPXDYES A SYSTEM DUMP WOULD BE TAKEN
*    UEPXNO A SYSTEM DUMP WOULD NOT BE TAKEN
*
* UEPXDTRM ADDRESS OF A 1-BYTE SHUTDOWN SETTING. EITHER:
*    UEPXDYES THE CICS SYSTEM WILL SHUTDOWN
*    UEPXNO THE CICS SYSTEM WILL NOT SHUTDOWN
*
UEPXD MAX ADDRESS OF A 4-BYTE MAXIMUM DUMPS VALUE

UEPXD CNT ADDRESS OF A 4-BYTE CURRENT DUMPS VALUE

UEPXD TST ADDRESS OF 16-BYTE DUMP STATS AREA.

4 CONTIGUOUS FULLWORDS REPRESENT:

NUMBER OF TRANSACTION DUMPS TAKEN
NUMBER OF TRANSACTION DUMPS SUPPRESSED
NUMBER OF SYSTEM DUMPS TAKEN
NUMBER OF System DUMPS Suppressed

UEPX DDAE ADDRESS OF A 1-BYTE DAEOPTION SETTING. EITHER:

UEPX DYES THE DUMP IS ELIGIBLE FOR DAE SUPPRESSION
UEPX DNO THE DUMP WONT BE SUPPRESSED BY DAE

OUTPUT = REGISTER 15 - RETURN CODE (UERC NORM OR UERCMEA)

THE FOLLOWING FIELDS WHICH REPRESENT DUMP TABLE
SETTINGS MAY BE MODIFIED AND THE MODIFIED VALUES WILL
BE WRITTEN BACK INTO THE DUMP TABLE ENTRY FOR THE
CURRENT DUMP CODE. SEE THE CUSTOMIZATION GUIDE FOR
FURTHER INFORMATION.

UEPX DSCP
UEPX DTXN
UEPX SYS
UEPX TRM
UEPX MAX
UEPX DDAE

EXIT-NORMAL = RETURN (14,12),RC=UERC NORM (CONTINUE DUMP)
RETURN (14,12),RC=UERC BYP (SUPPRESS DUMP)

EXIT-ERROR = N/A

CHANGE ACTIVITY :

$MOD(DFH$XDRQ) COMP(PROGRAM) PROD(CICS/ESA):

PN= REASON REL YYMMD HDXIII : REMARKS
$Ø1= PN64292 41Ø 95Ø111 PS : MODULE CREATION

***********************************************************************
EJECT
***********************************************************************

SET UP THE GLOBAL USER EXIT ENVIRONMENT :-
* IDENTIFY THE USER EXIT POINT
* SET UP EQUATES FOR REGISTERS
* ........................................................................
*    DFHUEXIT TYPE=EP,ID=XDUREQC
    DFHUEXIT TYPE=XPIENV
    COPY DFHPGISY
    COPY DFHSMMCY
*
* ........................................................................

* *
* DFH$STOR DSECT
XDRQSAVE DS 18F
PGMNAME DS CL8
DS ØF
WTO1E DS CL(WTO1L)
DS ØF
WTO2E DS CL(WTO2L)
DFH$STOL EQU -*DFH$STOR
EXEC_KEY DS CL1
*
* ........................................................................
* MAIN LINE CODE STARTS HERE
* ........................................................................
* DFH$XDRQ CSECT
DFH$XDRQ AMODE 31
DFH$XDRQ RMODE ANY
USING *,R15
B AROUND
DC CL8'DFH$XDRQ'
DC CL8'&SYSDATE'
AROUND DS ØH
DROP R15
SAVE (14,12)       SAVE REGS
LR R12,R15       SET-UP BASE REGISTER
USING DFH$XDRQ,R12
*
LR R2,R1       SET UP ADDRESSABILITY TO
USING DFHUEPAR,R2 USER EXIT PARM LIST
*
L R1,UEPDUMPC       ADDRESS THE DUMP TYPE VALUE
CLC "CL6'SR0001',Ø(R1) IS THIS SR0001?
BNE EXIT0       ..NO, WE ARE DONE
*
L R10,UEPXSTOR      SET UP ADDRESSING FOR XIP PARM LIST
USING DFHSMMC_ARG,R10
L R13,UEPSTACK     ADDRESS KERNEL STACK
DFHSMMCX CALL, X
CLEAR,  
IN,  
FUNCTION(GETMAIN),  
GET_LENGTH(DFH$STOL),  
SUSPEND(NO),  
INITIAL_IMAGE(X'00'),  
STORAGE_CLASS(CICS),  
OUT,  
ADDRESS((R11)),  
RESPONSE(*),  
REASON(*)

USING DFH$STOR,R11
ST R11,Ø(R10)  SAVE ACQUIRED ADDRESS  
LA R10,4(R10)  ADDRESS 4 BYTE OFFSET  
DROP R10
*
L R1,UEPPROG  MOVE PROGRAM NAME  
MVC PGMNAME,Ø(R1)  
MVC WTO1E,WTO1
*
USING DFHPGIS_ARG,R1Ø
L R13,UEPSTACK  ADDRESS KERNEL STACK
DFHPGISX CALL,  
CLEAR,  
IN,  
FUNCTION(INQUIRE_PROGRAM),  
PROGRAM_NAME(PGMNAME),  
OUT,  
EXECUTION_KEY(EXEC_KEY),  
RESPONSE(*),  
REASON(*)
*
CLI PGIS_RESPONSE,PGIS_OK CHECK RESPONSE  
BE DPGIS_OK  
MVC WTO1E+41(11),=CL10'INQ FAILED'  
B DPGIS_OK  
CLI EXEC_KEY,PGIS_USER  
BE DPGIS_OK  
MVC WTO1E+41(8),=CL8'CICS KEY'

DPGIS_OK DS ØH
*
LA R13,XDRQSAVE  SET UP SAVE AREA
*
L R1,UEPTRANID  MOVE TRANS-ID  
MVC WTO1E+13(4),Ø(R1)  
L R1,UEPUSER  MOVE USER-ID  
MVC WTO1E+18(8),Ø(R1)  
L R1,UEPTERM  MOVE TERM-ID  
MVC WTO1E+27(4),Ø(R1)  
MVC WTO1E+32(8),PGMNAME
* LA R1,WTO1E
WTO MF=(E,(R1))
CLI PGIS_RESPONSE,PGIS_OK CHECK RESPONSE
BNE EXIT
CLI EXEC_KEY,PGIS_CICS
BE EXIT
*
MVC WTO2E,WTO2
USING DFHPGIS_ARG,R1Ø
L R13,UEPSTACK ADDRESS KERNEL STACK
DFHPGISX CALL,
  CLEAR,
  IN,
  FUNCTION(SET_PROGRAM),
  PROGRAM_NAME(PGMNAME),
  EXECUTION_KEY(CICS),
  OUT,
  RESPONSE(*),
  REASON(*)
*
CLI PGIS_RESPONSE,PGIS_OK CHECK RESPONSE
BE EPGIS_OK
MVC WTO2E+21(13),=CL11'SET KEY ERROR'
EPGIS_OK DS ØH
MVC WTO2E+13(B),PGMNAME
LA R1,WTO2E
WTO MF=(E,(R1))
*
EXIT DS ØH
*
USING DFHSMMC_ARG,R1Ø
L R13,UEPSTACK ADDRESS KERNEL STACK
DFHSMMCX CALL,
  CLEAR,
  IN,
  FUNCTION(FREEMAIN),
  ADDRESS((R11)),
  STORAGE_CLASS(CICS),
  OUT,
  RESPONSE(*),
  REASON(*)
*
EXITØ DS ØH
L R13,UEPEPSA LOAD ADDRESS OF THE REG SAVE AREA
RETURN (14,12),RC=UERCNORM RESTORE REGS AND RETURN NORMAL
*
* DROP R2,R12
DS ØF
WTO1 WTO 'DFH$XDRQ TTTT UUUUUUUU TTTT PPPPPPPP USER KEY ',MF=L

CICS date simulator for year 2000 testing – part 2

This month we conclude the code for a CICS date simulator that allows year 2000 testing to be conducted without changing the underlying MVS system date.

```assembly
DC PL2'07',PL2'31',PL2'181'
DC PL2'08',PL2'31',PL2'212'
DC PL2'09',PL2'30',PL2'243'
DC PL2'10',PL2'31',PL2'273'
DC PL2'11',PL2'30',PL2'304'
DC PL2'12',PL2'31',PL2'334'
DC XL4'FFFFFFFF'

MSG00 DC CL24'Welcome to the simulator'
MSG01 DC CL24'DD not valid '
MSG02 DC CL24'MM not 01 to 12'
MSG03 DC CL24'Not DD/MM/YYYY or STOP'
MSG04 DC CL24'Simulator turned off'
MSG05 DC CL24'Simulator not active'
MSG06 DC CL24'Year not 1900 to 2099'
MSG07 DC CL24'No 19th century support'
MSG08 DC CL24'One day backwards'
MSG09 DC CL24'No 22nd century support'
MSG10 DC CL24'One day forwards'
```
MSG11 DC CL24'Welcome to year yyyy!'
MSG12 DC CL12'A leap year.'
MESS1 DC CL48'HDATØØ1I Date simulation started for DD/MM/YYYY.'
MESS2 DC CL36'HDATØØ2I Date simulation terminated.'
LTORG
END

MAP MST2000

MAPSET DFHMSD TYPE=&SYSPARM,MODE=INOUT,CTRL=FREEKB,LANG=ASM, *
       TIOAPFX=YES
MSY2ØØ Ø DFHMDI SIZE=(24,8Ø),LINE=1,COLUMN=1,MAPATTS=(COLOR)
       DFHMDF POS=(Ø1,23),LENGTH=32,COLOR=TURQUOISE, *
       INITIAL='DATE SIMULATION',
       DFHMDF POS=(Ø4,1),LENGTH=28,COLOR=NEUTRAL, *
       INITIAL='Enter DD/MM/YYYY or STOP ==>'
       DFHMDF POS=(Ø4,3Ø),LENGTH=1Ø,ATTRB=(IC,FSET),COLOR=YELLOW
       INCOMD DFHMDF POS=(Ø4,41),LENGTH=24,ATTRB=(ASKIP,BRT),COLOR=RED
       COMDERR DFHMDF POS=(Ø4,66),LENGTH=12,ATTRB=(ASKIP,BRT),COLOR=RED
       LEAP DFHMDF POS=(Ø5,41),LENGTH=1Ø,ATTRB=(IC,FSET),COLOR=YELLOW,
       INITIAL='Today is a'
       TODAY DFHMDF POS=(Ø5,52),LENGTH=9,ATTRB=(ASKIP,BRT),COLOR=YELLOW
       DFHMDF POS=(Ø7,1),LENGTH=34,COLOR=NEUTRAL, *
       INITIAL='Press PF7 to go one day backwards,'
       DFHMDF POS=(Ø7,36),LENGTH=27,COLOR=NEUTRAL, *
       INITIAL='PF8 to go one day forwards.'
       DFHMDF POS=(Ø9,1),LENGTH=35,COLOR=BLUE, *
       INITIAL='The following formats are available'
       DFHMDF POS=(Ø9,37),LENGTH=35,COLOR=BLUE, *
       INITIAL='from the FORMATTIME EXEC Command. '
       VARØ DFHMDF POS=(11,8),LENGTH=5,COLOR=NEUTRAL,INITIAL='YYDDD'
       VAR1 DFHMDF POS=(11,15),LENGTH=5,COLOR=BLUE,INITIAL='YYDDD'
       VAR2 DFHMDF POS=(11,24),LENGTH=7,COLOR=NEUTRAL,INITIAL='YYYYYDDD'
       VAR3 DFHMDF POS=(11,34),LENGTH=7,COLOR=BLUE,INITIAL='YYYYYDDD'
       VAR4 DFHMDF POS=(12,8),LENGTH=6,COLOR=NEUTRAL,INITIAL='YMMMDDD'
       VAR5 DFHMDF POS=(12,15),LENGTH=6,COLOR=BLUE,INITIAL='YMMMDDD'
       VAR6 DFHMDF POS=(12,24),LENGTH=8,COLOR=NEUTRAL,INITIAL='YYYYYMMMDDD'
       VAR7 DFHMDF POS=(12,34),LENGTH=8,COLOR=BLUE,INITIAL='YYYYYMMMDDD'
       VAR8 DFHMDF POS=(13,8),LENGTH=6,COLOR=NEUTRAL,INITIAL='YDDDMMDDD'
       VAR9 DFHMDF POS=(13,15),LENGTH=6,COLOR=BLUE,INITIAL='YDDDMMDDD'
       VAR10 DFHMDF POS=(13,24),LENGTH=8,COLOR=NEUTRAL,INITIAL='YYYYDDDDDMMDDD'
       VAR11 DFHMDF POS=(13,34),LENGTH=8,COLOR=BLUE,INITIAL='YYYYDDDDDMMDDD'
       VAR12 DFHMDF POS=(14,8),LENGTH=6,COLOR=NEUTRAL,INITIAL='DDMDYYYY'
       VAR13 DFHMDF POS=(14,15),LENGTH=6,COLOR=BLUE,INITIAL='DDMMDYYYY'
       VAR14 DFHMDF POS=(14,24),LENGTH=8,COLOR=NEUTRAL,INITIAL='DDMMDYYYY'
       VAR15 DFHMDF POS=(14,34),LENGTH=8,COLOR=BLUE,INITIAL='DDMMDYYYY'
       VAR16 DFHMDF POS=(15,8),LENGTH=6,COLOR=NEUTRAL,INITIAL='MMDDYYYY'
       VAR17 DFHMDF POS=(15,15),LENGTH=6,COLOR=BLUE,INITIAL='MMDDYYYY'
       VAR18 DFHMDF POS=(15,24),LENGTH=8,COLOR=NEUTRAL,INITIAL='MMDDYYYY'
       VAR19 DFHMDF POS=(15,34),LENGTH=8,COLOR=BLUE,INITIAL='MMDDYYYY'

DFHMDF POS=(17,1),LENGTH=34,COLOR=BLUE, *
INITIAL='YYYY Formats are only available at' *
DFHMDF POS=(17,36),LENGTH=34,COLOR=BLUE, *
INITIAL='CICS Version 4.1.0 or higher. ' *
DFHMDF POS=(19,1),LENGTH=34,COLOR=BLUE, *
INITIAL='This date simulator will not alter' *
DFHMDF POS=(19,36),LENGTH=42,COLOR=BLUE, *
INITIAL='any dates that are not obtained from CICS.' *
DFHMDF POS=(20,1),LENGTH=34,COLOR=BLUE, *
INITIAL='This includes dates obtained using' *
DFHMDF POS=(20,36),LENGTH=43,COLOR=BLUE, *
INITIAL='MVS SVCs or illegal COBOL verbs under CICS.' *

TIME DFHMDF POS=(24,1),LENGTH=5,COLOR=NEUTRAL,INITIAL='TIME:'
DATE DFHMDF POS=(24,7),LENGTH=8,COLOR=BLUE,INITIAL='XX.XX.XX'
APPL DFHMDF POS=(24,16),LENGTH=5,COLOR=NEUTRAL,INITIAL='DATE:'
APPL DFHMDF POS=(24,22),LENGTH=10,COLOR=RED,INITIAL='XX.XX.XXXX'
APPL DFHMDF POS=(24,64),LENGTH=7,COLOR=NEUTRAL,INITIAL='APPLID:'
APPL DFHMDF POS=(24,72),LENGTH=8,COLOR=BLUE

SYX2000 MACRO
* Module Name = SYX2000
* Command call EXIT to simulate year 2000
  DFHUEXIT TYPE=EP,ID=XEIOUT

******************************************************************************
* Register Equates
******************************************************************************
R1 EQU 1
R2 EQU 2
R7 EQU 7
R8 EQU 8
R9 EQU 9
DFHEIBR EQU 10
R11 EQU 11
R12 EQU 12
R13 EQU 13
R14 EQU 14
R15 EQU 15
EXIT_RC EQU R15
*
COPY DFHEIBLK
*
SYX2000 CSECT
SYX2000 AMODE 31
SAVE (14,12) Save registers
LR R11,R15
USING SYX2000,R11 Base register
LR R2,R1
USING       DFHUEPAR,R2       Point to exit parameter list
ICM         DFHEIBR,15,UEPEXECB  Point to EIB
BZ          RCVNORMAL             Exit as no EIB
USING       DFHEIBLK,DFHEIBR     Map EIB DSECT

ZAPDAT    DS          ØH
ICM         R7,15,UEPGAA       Address global area
BZ          RCVNORMAL           None - so leave
MVC         EIBDATE,Ø(R7)       Set date from global
ICM         R9,15,UEPARG       Point to command parameter list
BZ          RCVNORMAL           None - so leave
ICM         R8,15,Ø(R9)        Address Arg0
BZ          RCVNORMAL           None - so leave
CLC          Ø(2,R8),=XL2'4AØ2'  Asktime/abstime?
BNE         RCVNORMAL           No - so exit
ICM         R8,15,4(R9)        Address Arg1 ABSTIME address
BZ          RCVNORMAL           None - so leave
AP          Ø(8,R8),4(8,R7)    Add abstime offset

RCVNORMAL  DS          ØH
LA          EXIT_RC,UERCNORM   Set the return code to NORMAL
L           R13,UEPEPSA
RETURN (14,12),RC=(15)
LTORG
END         SYX2ØØØ

SY2PLT MACRO

SY2PLT         RMODE ANY
*
* PROGRAM       :SY2PLT
* DESCRIPTION   :This module changes the date seen by applications by
*                 enabling exit SYX2ØØØ. It runs in the PLTPI and reads a date
*                 from the CICS override parameter INITPARM=(SY2PLT=dd/mm/yyyy).
*                 This is verified and passed to SYX2ØØØ, the XEIOUT exit, in its
*                 global work area.
*
R2        EQU     2           Used by TRT instruction
EIBREG    EQU     3           EIB REG
DATAREG   EQU     4           DATA REG
BASE1     EQU     5           Base register
R6        EQU     6           Exit global area pointer
R7        EQU     7           Work register
DFHEISTG  DSECT
ATIME     DS      PL8         Absolute time
YEAR      DS      CL1Ø
YEARLEN   DS      H
DAYCNT    DS      F
HEXDATE   DS      F
COMDATE   DS      PL1Ø
DIVDATE   DS      PL9
DAYCNTP   DS      D

DAYCNTQ DS PL10
YRDIFF DS PL2
DYDIFF DS PL2
MMWK DS PL2
DDWK DS PL2
MESSØ DS CL60
LEAPIND DS X
DLENG DS H
EISTGEND EQU *

SY2PLT DFHEIENT CODEREG=(BASE1),
           EIBREG=(EIBREG),
           DATAREG=(DATAREG)

B BEGIN
DC CL12'PROGRAM ID: '
DC CL8'SY2PLT'
DC CL4':'
DC CL24'ASSEMBLY TIME AND DATE:'
DC CL8'&SYSTIME'
DC CL8'&SYSDATE'
BEGIN DS ØH

* Input parameter validation section
*

MVC MESSØ,MESS2
EXEC CICS WRITE OPERATOR TEXT(MESSØ)
EXEC CICS ASSIGN INITPARM(YEAR) INITPARMLEN(YEARLEN)
CLC YEARLEN,-H'0'
BE RETURN1 No - just return
MVC MESSØ,MESS3
Default error message
CLC YEARLEN,-H'10'
BNE INERR No
TRT YEAR(2),TRANTAB1 Numeric DD?
BNZ INERR No
CLI YEAR+2,C'/'
BNE INERR
TRT YEAR+3(2),TRANTAB1 Numeric MM?
BNZ INERR No
CLI YEAR+5,C'/'
BNE INERR
TRT YEAR+6(4),TRANTAB1 Numeric YYYY?
BNZ INERR No
PACK DDWK,YEAR(2)
CP DDWK,=PL1'0'
BE DDERR
ZAP DYDIFF,DDWK
PACK MMWK,YEAR+3(2)
LA R7,MONTAB

MONLOOP DS ØH
CLI Ø(R7),X'FF'
BE MMERR
CP MMWK,Ø(2,R7)

BE MONMATCH
LA R7,6(R7) No - try next entry
B MONLOOP

MONMATCH DS ØH
CP DDWK,2(2,R7) Max days this month
BH DDERR Exceeded
AP DYZDIFF,4(2,R7) Add month contribution
ZAP HEXDATE,DYZDIFF
CLC YEAR+6(2),=C'19'
BNE VYRØ1
MVI HEXDATE,X'00'
ZAP YRDIFF,=P'0'
B VYRØ2

VYRØ1 DS ØH
CLC YEAR+6(2),=C'20'
BNE NRANGE
MVI HEXDATE,X'01'
ZAP YRDIFF,=P'100'

VYRØ2 DS ØH
MVC HEXDATE+1(1),YEAR+9
MVO HEXDATE+1(1),YEAR+8(1)
MVO YRDIFF+1(1),YEAR+9(1)
MVN YRDIFF(1),YEAR+8
ZAP COMDATE,YRDIFF
ZAP DIVDATE,YRDIFF For leap day calculation
MP COMDATE,=PL4'315360' ABSTIME year difference
MP COMDATE,=PL2'100' Prevent spec exception
MP COMDATE,=PL3'1000' Prevent spec exception
MVI LEAPIND,X'00' Leap year indicator
DP DIVDATE,=PL1'4' Divide year diff by 4
CP DIVDATE+8(1),=PL1'0' Remainder zero - so leap year
BNZ NOTLEAP
CP DIVDATE(8),=PL1'0' 1900 was not a leap year
BZ NOTLEAP
MVI LEAPIND,X'FF' Set leap year indicator
AP HEXDATE,=PL1'1' Increase EIBDATE by one
CLC YEAR+3(2),=C'02' After February?
BH NOTLEAP Include this year's leap day
SP HEXDATE,=PL1'1' Decrease EIBDATE by one
SP DIVDATE(8),=PL1'1' Else take a day off

NOTLEAP DS ØH
AP DIVDATE(8),DYDIFF Add in DD/MM contribution

NOTFWD DS ØH
SP DIVDATE(8),=PL1'1' Take today off!
MP DIVDATE(8),=PL3'86400' Tot days in milliseconds
MP DIVDATE(8),=PL3'1000' + leap day difference
CLI LEAPIND,X'00'
BNE EXITSTRT
CLC YEAR+3(2),=C'02' February?
BNE EXITSTRT
EXEC CICS ASKTIME ABSTIME(ATIME)
EXEC CICS FORMATTIME ABSTIME(ATIME) DAYCOUNT(DAYCNT)
L R7,DAYCNT
CVD R7,DAYCNTP
ZAP DAYCNTQ,DAYCNTP
SP DAYCNTQ,=PL1'1'  Last night, not tonight
MP DAYCNTQ,=PL3'86400'
MP DAYCNTQ,=PL3'1000'  Absolute time last midnight
SP COMDATE,DAYCNTQ
EXEC CICS ENABLE EXIT('XEIOUT') PROGRAM('SYX2000')             *
   * GALENGTH(12)
EXEC CICS EXTRACT EXIT PROGRAM('SYX2000') GASET(R6)            *
   * GALENGTH(DLENG)
MVC Ø(4,R6),HEXDATE
MVC 4(8,R6),COMDATE+2
EXEC CICS ENABLE PROGRAM('SYX2000') START
MVC MESSØ,MESS1  Simulator started message
MVC MESSØ+37(10),YEAR

* SEND MESSAGE TO CONSOLE
*
SENDMSG DS ØH
EXEC CICS WRITE OPERATOR TEXT(MESSØ)
*
* RETURN AND FINISH
*
RETURN DS ØH
EXEC CICS RETURN
RETURN1 DS ØH
MVC MESSØ,MESS4  Simulator not active message
B SENDMSG
*
* Error Messages
*
DDERR DS ØH
MVC MESSØ+32(MESSØ),MSG01
B SENDMSG
MMERR DS ØH
MVC MESSØ+32(MESSØ),MSG02
B SENDMSG
INERR DS ØH
MVC MESSØ+32(MESSØ),MSG03
B SENDMSG
NRANGE DS ØH
MVC MESSØ+32(MESSØ),MSG04
B SENDMSG
*
* Constants
SY2FETAB MACRO

SY2FETAB EQU *
 *
* Code transaction in position 1-4
* Code program in position 9-16
* Table must be terminated by X'FFFFFFFF'
 *

SY2FE MACRO

SY2FE  RMODE ANY
     TITLE 'SY2FE - Front end transactions for SY2000'

* Program    : SY2FE
* Description: Front end applications to ensure EIBDATE is set to
              correct value when using the year 2000 simulator.
              Loads table SY2FETAB.

R6        EQU     6                        Pointer to commarea
TABPTR    EQU     7                        Pointer to applid table
DATAREG   EQU     8                        Data register
EIBREG    EQU     9                        EIB register
BASE      EQU     10                       Program base register
COMREG    DSECT
DFHEISTG  DSECT
MESSØ     DS      ØCL6Ø
TERMDATA  DS      ØCL78
TRANSACT  DS      CL4
            DS      CL7
SYSTID    DS      CL3
            DS      CL64
PRGNAME   DS      CL8
EISTGEND  EQU     *
SY2FE     DFHEIENT CODEREG=(BASE),EIBREG=(EIBREG),DATAREG=(DATAREG)
B       AØØØ
DC      CL12'PROGRAM ID: '
DC      CL8'SY2FE  '
DC      CL24'ASSEMBLY TIME AND DATE:'
DC      CL8'&SYSTIME'
DC      CL8'&SYSDATE'
AØØØ  DS      ØH
EXEC  CICS LOAD                                                X
     PROGRAM('SY2FETAB')                                  X
     SET(TABPTR)
EXEC  CICS ASKTIME
RØØØ  DS      ØH
CLC     Ø(4,TABPTR),EIBTRNID                                Give control
BE      Z1ØØ
LA      TABPTR,16(TABPTR)                                   Next table entry
CLI     Ø(TABPTR),X'FF'                                    End of table?
BNE     RØØØ  No keep searching
MVC     TERMDATA,OUTMSG2                                   Not found error
MVC     TRANSACT,EIBTRNID                                   Put tran in message
EXEC  CICS SEND                                              X
     FROM(TERMDATA)                                        X
LENGTH(78)
ERASE
MVC MESS0,MESS6
MVC MESS0+9(4),EIBTRNID Put tran in console message
EXEC CICS WRITE OPERATOR TEXT(MESS0)
Z000 DS ØH
EXEC CICS RETURN
Z100 DS ØH
MVC PRGNAME(8),8(TABPTR)
L R6,DFHEICAP
USING COMREG,R6
EXEC CICS XCTL
PROGRAM(PRGNAME)
COMMAREA(COMREG)
LENGTH(EIBCALEN)
B Z000
*
* CONSTANTS
*
OUTMSG2 DS ØCL78
DC CL16'xxxx is not defi'
DC CL3Ø'ned to SY2FETAB. Please contac'
DC CL32't support.'
MESS6 DC CL6Ø'HDATØØ6E xxxx not defined to SY2FETAB.'
LTORG
END SY2FE

Mick Masters
Principal Consultant
Cap Gemini (UK) © Xephon 1998

Call for papers

Why not share your expertise and earn money at the same time? CICS Update is looking for JCL, macros, program code, etc, that experienced CICS users have written to make their life, or the lives of their users, easier. We will publish it (after vetting by our expert panel) and send you a cheque when the article is published. Articles can be of any length and can be sent or e-mailed to Robert Burgess at any of the addresses shown on page 2. Why not call now for a free copy of our Notes for contributors?
IBM has announced Expedite/CICS Version 4.3, which links a host to the IBM Global Services Information Exchange service. Functions include Year 2000 readiness, a user-specific purge period, definition of alias names, validation of receive criteria, forced purge, sending a distribution list, and help panels.

IBM has also announced improvements in the collecting and reporting functions of the CICS and IMS features in its upgraded and renamed Performance Reporter for MVS. TME 10 Performance Reporter for OS/390 includes seven performance features covering CICS, system, network, IMS, workstation, AS/400, and accounting, and four OS/2-based features for reporting, analysing, or planning resource usage.

For further information contact your local IBM representative.

***

Compuware has announced a new release of XPEDITER/CICS, its fullscreen, source-level testing and debugging product. XPEDITER/CICS maintains an audit trail of all production file changes made within the file utility. The logging facility enables organizations to keep track of any changes made within the XPEDITER/CICS file utility, including adds, updates, and deletes of dataset records, IMS segments, temporary storage records and queues, transient data records, and DB2 tables. The logging facility writes formatted data to a dataset. Output can be formatted to suit individual site requirements.

In addition, XPEDITER/CICS offers new functionality that enables users to gain control and test CICS programs that are started independently of a terminal.

Release 7.0 also offers XPEDITER command scripting, four-digit year displays, additional FIND command options, improved on-line help, and support for IBM CICS Transaction Server Release 1.2.

For further information contact:
Compuware, 31440 Northwestern Highway, PO Box 9080, Farmington Hills, MI 48334-2564.
Tel: (800) 737 7300.
Compuware, 163 Bath Road, Slough, Berks, SL1 4AA, UK.
Tel: (01753) 774000.

***

Netscape Communications has started shipping extensions for accessing BEA Tuxedo and IBM MQSeries, allowing integration with CICS and IMS systems.

New features in the application server include better scalability through load balancing features and end-to-end performance enhancements. Availability comes via distributed state/session management combined with fail-over, failure detection, and failure recovery capabilities. There are a range of management facilities geared to distributed servers and applications, and integration tools provide connectivity to existing applications and legacy systems.

For further information contact:
Netscape Communications, 501 East Middlefield Road, Mountain View, CA 94043, USA.
Tel: (650) 254 1900.

---

**CICS news**

---

xepphon