IBM Language Environment for z/VSE

Writing Language Environment Main Assembler Applications For CICS/TS on z/VSE

Revision : Fri, 1 August 2008

Table of Contents

Disclaimer

No warrently or support is implied or provided by IBM for any use by the reader of the information provided in this document. Its use is solely the readers responsilbility.

Fri, 1 August 2008
Garry Hasler
LE z/VSE Development
Australia Development Labratory for zSeries, West Perth, Australia

Overview

Starting with z/VSE 4.1, Language Environment for z/VSE 1.4.5 has provided support for LE-conforming main assembler programs under CICS, with the following restrictions (see the LE z/VSE Programming Guide) :

- The HANDLE LABEL option of any appropriate EXEC CICS commands *is not* used.
- The NOEPILOG and NOPROLOG CICS translator options are specified.
- If calling a HLL (High Level Language – eg COBOL/VSE) or LE assembler subroutine that will use CICS services from an LE assembler main routine, you should ensure that the appropriate CICS control blocks (DFHEIB, DFHCOMMAREA) are passed as parameters.
- Overlay programs are not used.
- The Assembler program is re-entrant.
- LE z/VSE Library Routine Retention is not used.

This documentation assumes HLASM V1.5 for z/VSE or above with the options CODEPAGE(047C), OPTABLE(UNI,NOLIST), NOCOMPAT, LANGUAGE(EN) and RENT set.

Register Usage

It is imperitive that correct register usage is maintained throughout the assembler program execution. On entry into the assembler main routine registers will contain the following values after they are passed through the CEEENTRY macro:

- **R0** Undefined
- **R1** Undefined
- **R2** Undefined
- **R13** Assembler routines LE conforming DSA (register save area)
- **R14** Return address
- **R15** Entry point address

On entry into an assembler routine, the caller's registers (R14 through R12) are saved into the DSA provided by the caller. After allocating a DSA (which sets the NAB field correctly in the new DSA), the first halfword of the DSA is set to zero and the backchain is set appropriately.

At all times while the assembler routine is running, R13 (DFHEIPLR) must contain the executing routine's DSA.

At any call (excluding CICS/TS HLAPI calls) and return point, R12 must contain the CAA address.

When exiting from an assembler routine the CEETERM macro will restore registers as appropriate.

Available Condition Handling

LE z/VSE default condition handling actions occur for assembler routines unless you have registered a user-written condition handler using CEEHDLR (see LE z/VSE Programming Guide for more information).

Use of the EXEC CICS HANDLE ABEND PROGRAM command is supported however use of the EXEC CICS ABEND LABEL should not be used in a main LE-conforming assembler program.

The use of EXEC CICS HANDLE CONDITION is supported under the provision that the programmer save DFHEIPLR prior to issuing the EXEC CICS command that may experience a condition into an unmodified register (eg R2). Then immediately restore DFHEIPLR upon entry to the handle routine (eg LR DFHEIPLR,R2).

Resource Availability

LE/VSE relinquishes all enclave-level resources that were obtained by LE/VSE when the enclave terminates, and all process-level resources when the CICS transaction completes.

Use of the PARMREG option on the CEEENTRY macro should not be used when running in a CICS environment.

Standard CICS/TS resource availablity also applies to LE-conforming main assembler routines.

CICS/TS Translator Considerations

The current CICS/TS translator on z/VSE does not directly support LE-conforming assembler main routines. However, by using the NOEPILOG and NOPROLOG options this will allow the programmer to provide the correct entry and exit assembler code to enable the LE assembler main program to execute.

Here is an example of LE-conforming assembler main entry code for use under CICS :

```
// EXEC DFHEAP1$
*ASM CICS(NOPROLOG NOEPILOG NOEDF)
          TITLE 'LE/ASM main program under CICS'
CICSASMM CEEENTRY PPA=ASMPPA,MAIN=YES,AUTO=STORLEN
          USING DFHEISTG,DFHEIPLR
```

The DFHEISTG macro is still required to mark the start of dynamic storage.

The following equates should be added to the usual assembler program register equates :

```
DFHEIPLR EQU   13
DFHEIBR  EQU   10
```

The following LE macros should also be included in the LE main CICS assembler program :

CEEPPA
CEEDSA
CEECAA

LE z/VSE Provided Assembler Macros

Under CICS all the available LE z/VSE assembler macros, except for the CEELOAD macro, can be used. In place of the CEELOAD macro it is recommended that the CEEFETCH and CEERELES macros be used instead.

Use of the CEETERM macro should be used at the complete termination of the program exit point inplace of a EXEC CICS RETURN. For pseduo-conversational programming use of the EXEC CICS RETURN TRANSID should be used at non-termination wait points.

Linkedit Considerations

The CICS/TS provided EXEC interface module DFHEAI needs to be included at linkedit time. The programmer should also ensure that the LE assembler main program CSECT is either set or specified as the module entry point. The linkedit map output should be reviewed to verify the correct linkedit settings prior to executing the application in an active CICS/TS system.

CEDF Considerations

When executing the CICS Execution Diagnositic Facility (CEDF) transaction on an LE assembler main routine you must specify the EDF option at CICS translation time.

During the execution of the LE assembler main under CEDF the message “REGISTER 13 DOES NOT ADDRESS DFHEISTG” will be displayed in the debugging session. This is expected and does not indicate a problem.

Use of PF5 “Working Storage” display will not function on an LE assembler main routine. This is because CICS/TS requires LE z/VSE to provide the address of LE applications working storage area. As there is no LE z/VSE event handler for assembler programs there is no service available to provide this information to CICS/TS.

Defining LE Assembler Main Programs to CICS/TS

CICS/TS with LE z/VSE is capable of determining that you have executed an assembler program. For those that wish to define their applications manually, or who do not use the IBM provided CICS/TS Autoinstall program exit(s), here are some CEDA settings to use for LE-conforming assembler :

```
OBJECT CHARACTERISTICS                                    CICS RELEASE = 0411
  CEDA  View PROGram( CICSASMM )
   PROGram        : CICSASMM
   Group          : LEASM
   DEscription    :
   Language       : Assembler          CObol | Assembler | C | Pli
   RELoad         : No                 No | Yes
   RESident       : No                 No | Yes
   USAge          : Normal             Normal | Transient
   USEsvacopy     : No                 No | Yes
   Status         : Enabled            Enabled | Disabled
   RSl            : 00                 0-24 | Public
   Cedf           : Yes                Yes | No
   DAtalocation   : Any                Below | Any
   EXECKey        : User               User | Cics
```

Assembler CICS/TS Main Program Example

Following is an example LE-conforming assembler main program for use under CICS. The source example does not include any subroutines or called modules. The provided source code is intended more for reference purposes than as a functional example.

The example shows use of LE callable services (CEE5DLY, CEEMOUT), the CEEFETCH and CEERELES macros and HANDLE CONDITION use under CICS/TS. It is not pseudo-conversational.

The program being fetched in the example can be replaced with any LE-conforming HLL or LE assembler subroutine desired.

Here is some JCL that can be used to assemble and linkedit the following sample program assuming the source code is stored in ICCF library 10 as member LECICASM.

```
* $$ JOB JNM=LECICAS,DISP=D,CLASS=A
* $$ LST DISP=D,CLASS=Q,PRI=3
* $$ PUN DISP=I,DEST=*,PRI=9,CLASS=A
// JOB LECICAS TRANSLATE PROGRAM LECICASM
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
$ $$ LST DISP=D,CLASS=Q,PRI=3
// JOB LECICAS - ASSEMBLE PROGRAM CICSASMM
// SETPARM CATALOG=1
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE,CATALOG=PRIMARY.$$C
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD1.BASE)
// OPTION ERRS,SXREF,SYM,CATAL,NODECK
   PHASE CICSASMM,*
   INCLUDE DFHEAI
/. ENDCAT
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
               -200K,ABOVE),RENT'
* $$ END
// ON $CANCEL OR $ABEND GOTO ENDJ2
// OPTION NOLIST,NODUMP,DECK
// EXEC DFHEAP1$,SIZE=512K
*ASM XOPTS(CICS,NOEPILOG,NOPROLOG,NOEDF)
* $$ SLI ICCF=(LECICASM),LIB=(0010)
/*
/. ENDJ2
// EXEC IESINSRT
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKEDT,SIZE=256K
/. NOLNK
#&
$ $$ EOJ
* $$ END
/&
* $$ EOJ
```

```
CICSASMM CEEENTRY PPA=ASMPPA,MAIN=YES,AUTO=STORLEN
         USING DFHEISTG,DFHEIPLR
         XR    R4,R4
         LR    R2,DFHEIPLR                save R13 incase handle done
         EXEC CICS SEND CONTROL ERASE FREEKB
         EXEC CICS ASKTIME
         EXEC CICS SEND TEXT FROM(CEEDLY_MSG) LENGTH(46) WAIT FREEKB
         LA   R1,2
         ST   R1,DELAY
         LA   R1,DELAY
         ST   R1,A_DELAY
         LA   R6,FBCODE
         ST   R6,A_FBC
         LA   R1,A_DELAY
         L    R15,=V(CEE5DLY)
         BALR R14,R15
         CLC  FBCODE,CEE000          Did call work ok?
         BNE  Exit_stage_right
         EXEC CICS SEND CONTROL ERASE FREEKB
         EXEC CICS SEND TEXT FROM(HANDLE_MSG) LENGTH(48) WAIT FREEKB
         EXEC CICS DELAY FOR SECONDS(1)
         EXEC CICS HANDLE CONDITION QIDERR(HANDLED)
         EXEC CICS READQ TS QUEUE(NOTHERE) INTO(TSQDATA) LENGTH(TSQL)
DODCALL  DS    0H
         CALL  CEEMOUT,(CVSEMSG,DEST,0)   issue dynamic call message
         MVC   CSUBNAME,=CL8'CUCICS  '  setup name of dynamic routine
         LA    R4,HLLNAME               address parm list for LE macro
         EXEC CICS IGNORE CONDITION QIDERR
         MVC   FETCHPL(l'FETCH_PL),FETCH_PL    copy in CEEFETCH constants
         ST    R4,FETCHPL                prepare parameter list contents
         LA    R4,FETCH_TOKEN
         ST    R4,FETCHPL+8
         LA    R4,FBCODE
         O     R4,=X'80000000'           indicate last parm
         ST    R4,FETCHPL+12
FETCH1   CEEFETCH MF=(E,FETCHPL)         fetch module
         ST    R15,LOADADDR               save load addr
         XR    R14,R14
         LA    R14,15
         STH   R14,CONSMSGL
         LA    R14,CONSMSGL
         ST    R14,A_MSGLEN
         LA    R14,FTCH_MSG
         ST    R14,A_MSGTXT
         LA    R1,C_PARMS                 Get parm list addr in R1
         LTR   R15,R15                    is the program AMODE31?
         BZ    Exit_stage_right           if fetch failed, exit
         BNM   NOTXA                      no, dont use BASSM
         BASSM 14,15                      execute routine
         AMODESW SET,AMODE=(14)           reset to our AMODE
         B     DYNEND                     time to end
NOTXA    EQU   *
         CALL  CEEMOUT,(NONXA,DEST,0)     issue non-xa call message
         L     15,LOADADDR                ready branch address
         BALR  14,15                      call program using 370 instr
DYNEND   DS    0H
         MVC   RELESPL(l'RELES_PL),RELES_PL
         LA    R4,FETCH_TOKEN
         ST    R4,RELESPL
         LA    R4,FBCODE
         O     R4,=X'80000000'           indicate last parm
```

```
         ST    R4,RELESPL+4
         CEERELES MF=(E,RELESPL)
         CLC  FBCODE,CEE000        Did call work ok?
         BNE  Exit_stage_right
         CALL  CEEMOUT,(RELSMSG,DEST,0)    say ceereles worked ok.
         B     Exit_stage_right
HANDLED  DS   0H
         LR   DFHEIPLR,R2                 restore R13 from R2
         EXEC CICS SEND CONTROL ERASE FREEKB
         EXEC CICS SEND TEXT FROM(HANDLED_MSG) LENGTH(42) WAIT FREEKB
         B    DODCALL
Exit_stage_right   DS   0H
         EXEC CICS SEND TEXT FROM(FINI_MSG) LENGTH(44) ERASE FREEKB
         CEETERM                          All done, return to LE/VSE
* Constants
DEST     DC   F'2'
CEE000   DC   XL12'00'
NOTHERE  DC   C'NOTHERE '
CEEDLY_MSG  DC   C'About to call CEE5DLY under CICS for 2 seconds'
HANDLE_MSG  DC   C'About to issue EC READQ TS for non-existant TS q'
HANDLED_MSG DC   C'TSQ read failure has been handled!! Yay!!!'
FINI_MSG    DC   C'Assembler main under CICS testcase complete.'
MSG_LEN  DC  H'15'
FTCH_MSG DC  CL80'CUCICS FINISHD',X'00'
*
CVSEMSG  DC    Y(CVSELEN)
CVSETXT  DC    C'ASMLEPRG - Using CEEFETCH to load C/VSE Subroutine'
CVSELEN  EQU   *-CVSETXT
*
NONXA    DC    Y(NONXAL)
NONXAT   DC    C'ASMLEPRG - Calling Fetched subroutine via BALR'
NONXAL   EQU   *-NONXAT
*
RELSMSG  DC    Y(RELSLEN)
RELSTXT  DC    C'ASMLEPRG - C/VSE Subroutine successfully released.'
RELSLEN  EQU   *-RELSTXT
*
FETCH_PL CEEFETCH SCOPE=ENCLAVE,MF=L
FETCH_PL_LEN  EQU *-FETCH_PL
RELES_PL CEERELES MF=L
RELES_PL_LEN  EQU *-RELES_PL
ASMPPA   CEEPPA
         CEEDSA
         CEECAA
         DFHEISTG                         Extended save area for CICS
*   Paramter list to pass to LE for the CEEFETCH macro
         DS    0D
HLLNAME  DS    0CL10
LEPREFIX DS    H
CSUBNAME DS    CL8
FETCH_TOKEN  DS  F
FBCODE   DS    3F
*
FETCHPL  DS    CL(FETCH_PL_LEN)
         DS    0F
RELESPL  DS    CL(RELES_PL_LEN)
*
CONSMSGL DS    H
DELAY    DS    F
A_DELAY  DS    A
A_FBC    DS    A
```

```
TSQDATA  DS    CL256
TSQL     DS    F
CALLMOUT CALL  ,(,,),VL,MF=L
LOADADDR DS    F
*     CUCICS PARMS
C_PARMS  DS    0F
A_MSGLEN DS    F
A_MSGTXT DS    F
STORLEN  EQU   *-DFHEISTG
         COPY  DFHEIBLK
R0       EQU   0
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R12      EQU   12
R14      EQU   14
R15      EQU   15
DFHEIPLR EQU   13
DFHEIBR  EQU   10
         END   CICSASMM
```

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

CICS, CICS/TS, IBM, Language Environment, VSE/ESA, z/VSE

Other company, product, or service names, may be the trademarks or service marks of others.

Comments and Questions

Comments or questions on this documentation are welcome. Please send your comments to:
zvse@de.ibm.com