

FOCUS Version 7.0
 July 30, 1999

TM7950

Using VM HLI With LE

The Host Language Interface (HLI), in use since the early 1980's, allows users to write programs that read and write FOCUS databases in languages such as PL/I, FORTRAN, COBOL, and Assembler. In prior releases, the HLI environment was initialized prior to calling any user-written HLI program, and the user program was called as a subroutine. If the subroutine required IBM's Language Environment (LE), this implementation prevented CMS from obtaining the information needed to correctly initialize that environment; therefore, LE was not supported.

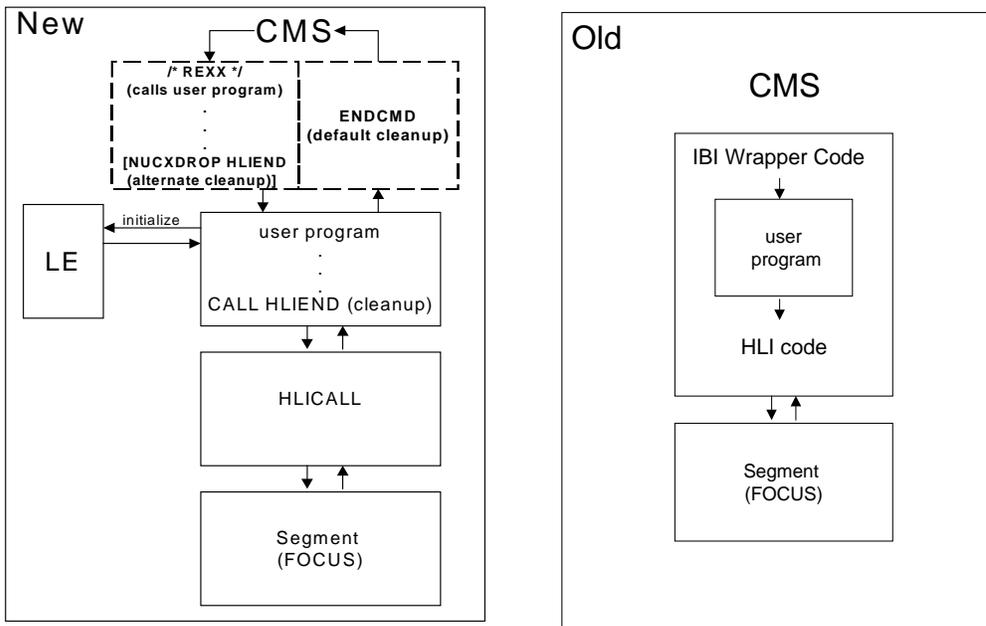
A new HLI implementation is now available that fully supports LE. With this new implementation, you code the user-written data access program as a main program, and it calls the HLI environment as a subroutine (named HLICALL). Because HLI no longer takes control prior to invoking the user program, if the user program is LE-enabled CMS can obtain the information needed to initialize the LE environment.

Note: The user-written HLI module does not have to be LE-enabled in order to use HLICALL; HLICALL works with non-LE environments. However, modules that use LE *must* use HLICALL.

With the new implementation, user modules are relocatable without the entry point resetting required in prior implementations; therefore, HLICALL does not use the GENHLI and IUGUSER EXECs needed previously to initialize the HLI environment.

An additional advantage to this implementation is that certain types of user programs, such as PL/I programs, may be prone to storage leaks when called as subroutines; this implementation solves that problem. It also eliminates errors associated with passing incorrect parameter lists because CMS now handles passing parameter lists.

The following diagram illustrates the two HLI implementations:



How to Generate and Link a User HLI Program With LE

The following guidelines apply when creating an LE-enabled module for use with HLICALL:

1. Code your program with MAIN option, if available in the language you use to write the program. For example, Assembler does not support the MAIN option, PL/I does. Some languages, such as C, may require you to execute an EXEC after compiling but prior to loading.
2. Issue any GLOBAL TXTLIB commands needed for your environment.
3. Issue any GLOBAL LOADLIB commands needed for your environment.
4. Compile the program using the LE options needed with the relevant compiler. For example, a typical requirement is the following GLOBAL TXTLIB command:

```
GLOBAL TXTLIB SCEELKED
```

5. Issue the LOAD command:

```
LOAD user_hli_program [HLICALL] (le_load_options)
```

where:

user_hli_program

Is the name of your user-written HLI program.

HLICALL

Is needed only if you are linking HLICALL statically. For dynamic linkage, omit HLICALL from the syntax and follow the instructions in How to Link COBOL Programs That Call Subroutines Dynamically.

le_load_options

Are LOAD options needed for LE based on the programming language. It is your responsibility to add the correct options. These may include:

AMODE 31

RMODE ANY

RLDSAVE

MAP

6. Issue the GENMOD command to generate your module:

```
GENMOD user_hli_module (le_options)
```

where:

user_hli_module

Is the name of your user-written HLI module.

le_options

Are any LE options needed for your programming environment. For example, a PL/I module would need the option FROM PLISTART. Other options may include:

FROM CEESTART

AMODE 31

RMODE ANY

How to Link COBOL Programs That Call Subroutines Dynamically

Some COBOL programs are generated to call subroutines dynamically, although LE is not designed to support this type of call. If your program uses this type of dynamic linkage, you have two options:

Change your calls to FOCUS so that you call HLICALL instead:

```
WORKING-STORAGE SECTION.  
01 FOCUS PIC X(8) VALUE 'HLICALL '.
```

Rename or copy the HLICALL TEXT file to FOCUS TEXT:

```
RENAME HLICALL TEXT A FOCUS TEXT A
```

or

```
COPYFILE HLICALL TEXT A FOCUS TEXT A (OLDDATE REPLACE
```

De-Initializing the HLICALL Environment

There are three choices for de-initializing the HLI environment after your HLI module executes:

1. The user-written program can issue the CALL HLIEND command provided by HLICALL. This is the recommended de-initialization method.
2. If you do not CALL HLIEND in your program and your program is called from an EXEC, you can use the following command in the EXEC:

```
NUCXDROP HLIEND
```

This allows existing programs to be reloaded.

3. The HLI initialization procedure defines an ENDCMD command. If you do not explicitly de-initialize the HLI environment using either of the first two techniques, the ENDCMD will attempt de-initialization. However, this de-initialization will not be as complete as the de-initialization accomplished by CALL HLIEND or NUCXDROP HLIEND. Files being written to disk using OS services such as HLIPRINT may also be lost.

Sample Programs Using HLICALL

Following are two sample programs, one COBOL and one PL/I. For each program, a sample REXX EXEC is included.

The first call to HLI, no matter what it is, initializes the HLI environment. It initializes the environment even if the CALL fails.

The call to HLIEND cleans up the environment.

REXX EXEC for COBOL HLI Program

```
/*  
   THIS REXX EXEC WILL DO A TEST OF A COBOL HLI PROGRAM  
   USING A STATIC CALL TO FOCUS(HLICALL)  
*/  
TRACE A  
"EXEC VMLINK COBOL120 "  
"GLOBAL TXTLIB SCEELKED "  
"GLOBAL LOADLIB SCEERUN "  

```

COBOL HLI Program

IDENTIFICATION DIVISION.			TM700010
PROGRAM-ID. TM7950S .			TM700020
ENVIRONMENT DIVISION.			TM700030
CONFIGURATION SECTION.			TM700040
SOURCE-COMPUTER. IBM.			TM700050
OBJECT-COMPUTER. IBM.			TM700060
INPUT-OUTPUT SECTION.			TM700070
FILE-CONTROL.			TM700080
DATA DIVISION.			TM700090
FILE SECTION.			TM700100
WORKING-STORAGE SECTION.			TM700110
01 FOCUS		PIC X(8) VALUE 'HLICALL '.	TM700120
01 HLIEND		PIC X(8) VALUE 'HLIEND '.	TM700130
01 CMDS.			TM700140
05 OPN		PIC X(4) VALUE 'OPN '.	TM700150
05 CLO		PIC X(4) VALUE 'CLO '.	TM700160
05 SHO		PIC X(4) VALUE 'SHO '.	TM700170
05 FST		PIC X(4) VALUE 'FST '.	TM700180
05 NEX		PIC X(4) VALUE 'NEX '.	TM700190
05 INFO		PIC X(4) VALUE 'INFO'.	TM700200
01 FCB.			TM700210
05 FILENAME		PIC X(8) VALUE 'CAR '.	TM700220
05 FILETYPE		PIC X(8) VALUE 'FOCUS '.	TM700230
05 FILEMODE		PIC X(4) VALUE 'A1 '.	TM700240
05 SUIND		PIC X(4) VALUE ' '.	TM700250
05 FILLER1		PIC X(8) VALUE SPACES.	TM700260
05 SINKID		PIC X(8) VALUE ' '.	TM700270
05 FILLER2		PIC X(28) VALUE SPACES.	TM700280
05 FCB-ECHO		PIC X(4) VALUE 'ECHO'.	TM700290
05 FILLER3		PIC X(20) VALUE SPACES.	TM700300
05 STAT-CD	COMP	PIC S9(5) VALUE +0.	TM700310
05 FILLER4		PIC X(104) VALUE SPACES.	TM700320
01 NAMES-AREA.			TM700330
05 FIELD-0001		PIC X(12) VALUE 'COUNTRY '.	TM700340
05 FIELD-0002		PIC X(12) VALUE 'CAR '.	TM700350
05 FIELD-0003		PIC X(12) VALUE 'MODEL '.	TM700360
05 FIELD-0004		PIC X(12) VALUE 'BODY '.	TM700370
01 NUMB0	COMP	PIC S9(9) VALUE 0.	TM700380
01 NUMB1	COMP	PIC S9(9) VALUE 1.	TM700390
01 NUMB2	COMP	PIC S9(9) VALUE 2.	TM700400
01 NUMB3	COMP	PIC S9(9) VALUE 3.	TM700410
01 NUMB4	COMP	PIC S9(9) VALUE 4.	TM700420
01 NNULL		PIC X(8) VALUE SPACE.	TM700430
01 NREPEAT	COMP	PIC S9(9) VALUE 1.	TM700440
LINKAGE SECTION.			TM700450
PROCEDURE DIVISION.			TM700460
MAIN SECTION.			TM700470
CALL FOCUS USING OPN FCB NUMB0			TM700480
IF STAT-CD NOT EQUAL 0			TM700490
DISPLAY 'BAD OPEN OF FOCUS FILE '			TM700500
GO TO MAIN-EXIT.			TM700510
CALL FOCUS USING SHO FCB NAMES-AREA NUMB1.			TM700520
IF STAT-CD NOT EQUAL 0			TM700530
DISPLAY 'BAD SHOW OF FOCUS FILE '			TM700540
GO TO MAIN-EXIT.			TM700550
CALL FOCUS USING CLO FCB.			TM700560
IF STAT-CD NOT EQUAL 0			TM700570
DISPLAY 'BAD CLOSE OF FOCUS FILE'.			TM700580
MAIN-EXIT. EXIT.			TM700590
CALL HLIEND.			TM700600
MOVE STAT-CD TO RETURN-CODE.			TM700610
GOBACK.			TM700620

Sample PL/I HLI Program

The following example includes a REXX EXEC and PL/I program to use with HLICALL.

REXX EXEC for PL/I HLI Program

```
/*
  THIS REXX EXEC WILL DO A TEST OF A PL/I HLI PROGRAM
  USING A STATIC CALL TO FOCUS(HLICALL) AS PER TM7950
*/
TRACE A
"EXEC VMLINK PL1 "
"GLOBAL MACLIB SCEEMAC SCEESAMP PLICOMP "
"GLOBAL TXTLIB SIBMCALL SCEELKED SCEEOBJ PLILIB CMSLIB "
"GLOBAL LOADLIB SCEERUN SCEEILBO "
"PLIOPT TM7950PL "
"LOAD TM7950PL HLICALL (AMODE 31 RMODE ANY RLDSAVE MAP "
"GENMOD TM7950PL ( SYSTEM FROM PLISTART "
"FILEDEF HLIPRINT DISK HLIPRINT DATA A (LRECL 88 RECFM FB "
"FILEDEF CAR DISK CAR FOCUS A "
"TM7950PL"
EXIT
```

PL/I HLI Program

```
TM7950P: PROC OPTIONS(MAIN);
DCL FOCUS ENTRY OPTIONS(ASSEMBLER,INTER);
DCL HLIEND ENTRY OPTIONS(ASSEMBLER,INTER);
DCL FIXBIN0 FIXED BIN(31) INIT(0);
DCL FIXBIN1 FIXED BIN(31) INIT(1);
DCL FIXBIN2 FIXED BIN(31) INIT(2);
DCL FIXBIN3 FIXED BIN(31) INIT(3);
DCL FIXBIN4 FIXED BIN(31) INIT(4);
DCL 1 FCB STATIC,
2 FCB_FILENAME CHAR(8) INIT('CAR '),
2 FCB_FILLER01 CHAR(8) INIT('FOCUS '),
2 FCB_FILLER01 CHAR(4) INIT('A '),
2 FCB_SU_FLAG CHAR(4),
2 FCB_PROC_NAME CHAR(8),
2 FCB_SYNC_NAME CHAR(8),
2 FCB_FILLER02 CHAR(20) INIT(' '),
2 FCB_RECORD_ADDR CHAR(8) INIT(' '),
2 FCB_HLIPRINT_FLAG CHAR(4) INIT('ECHO'),
2 FCB_PASSWORD CHAR(8),
2 FCB_SEG_NAME CHAR(8) INIT(' '),
2 FCB_SEG_NUM FIXED BIN(31),
2 FCB_RESULT_CODE FIXED BIN(31),
2 FCB_RECORDS_RETURNED FIXED BIN(31),
2 FCB_FILLER03 CHAR(28),
2 FCB_RECORD_LENGTH FIXED BIN(31),
2 FCB_ALL_RECORDS_LENGTH FIXED BIN(31),
2 FCB_FILLER04 CHAR(64);
DCL 1 NAMES_AREA STATIC,
2 FIELD_0001 CHAR(12) INIT('COUNTRY '),
2 FIELD_0002 CHAR(12) INIT('CAR '),
2 FIELD_0003 CHAR(12) INIT('MODEL '),
2 FIELD_0004 CHAR(12) INIT('BODY ');
DCL SYSPRINT FILE STREAM OUTPUT PRINT;
CALL FOCUS('OPN ',FCB, FIXBIN0);
CALL FOCUS('SHO ',FCB,NAMES_AREA, FIXBIN1);
CALL FOCUS('CLO ',FCB, FIXBIN0);
CALL HLIEND;
END; /* OF TM7950P*/
```