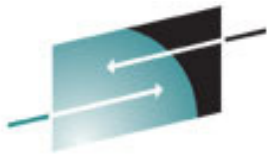
What's New in z/OS Language Environment?



John Monti
IBM Poughkeepsie
jmonti@us.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

- CICS®
- IMS
- Language Environment®
- OS/390®
- z/OS®

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

IEEE is a trademark in the United States and other countries of the Institute of Electrical and Electronics Engineers, Inc.

POSIX® is a registered Trademark of The IEEE.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

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

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Agenda

- New Function APARs
- What's New in z/OS V1.12?
- What's New in z/OS V1.11?

- Additional information available in Appendix:
  - Additional Reference Material
  - What's New in z/OS V1.10?
  - Sources for Additional Information

New Function APARS

- High Register Support
- pthread_yield_max()
- CEEPIPI call_sub with writable static

New Function APARs

- **PM04026 – High Register Support**
  - UK59090 (R10)
  - UK59091 (R11)
  - AMODE 31 CEEDUMP support to display full 64 bit registers
    - When unavailable the high half of the 64 bit register is displayed as '********'
  - AMODE 31 LEDATA support to display high halves of 64 bit registers when formatting the MCH control block

New Function APARs

- **PM04026 – High Register Support**
  - **CEEDUMP**

```
Machine State:

      ILC..... 0002    Interruption Code..... 0009

      PSW..... 078D2400 A19C60FE

      GPR0..... 00000000_00000000  GPR1..... 00000000_0000000A  GPR2.....
   00000000_A1CD09BC  GPR3..... 00000000_219C60B8

      GPR4..... 00000000_2199D2D8  GPR5..... 00000000_21F91A00  GPR6.....
   00000000_21F92AC8  GPR7..... 00000000_219BDE40

      GPR8..... 00000000_A19C63A8  GPR9..... 00000000_21F93368  GPR10....
   00000000_A19C6070  GPR11.... 00000000_A19C60A0

      GPR12.... 00000000_21713B58  GPR13.... 00000000_2199D6D8  GPR14....
   00000000_00000000  GPR15.... 00000000_00000006
```

New Function APARs

- **PM04026 – High Register Support**
  - **IPCS**

```
Machine State
+000248  MCH_EYE:ZMCH
+000250  GPR00:00000000    GPR01:0000000A
+000258  GPR02:A1CD09BC    GPR03:219C60B8
+000260  GPR04:2199D2D8    GPR05:21F91A00
+000268  GPR06:21F92AC8    GPR07:219BDE40
+000270  GPR08:A19C63A8    GPR09:21F93368
+000278  GPR10:A19C6070    GPR11:A19C60A0
+000280  GPR12:21713B58    GPR13:2199D6D8
+000288  GPR14:00000000    GPR15:00000006
+000290  PSW:078D2400 A19C60FE
```

New Function APARs

- **PM04026 – High Register Support**
  - **IPCS**

```
+000388  GPR_H00:00000000        GPR_H01:00000000
+000390  GPR_H02:00000000        GPR_H03:00000000
+000398  GPR_H04:00000000        GPR_H05:00000000
+0003A0  GPR_H06:00000000        GPR_H07:00000000
+0003A8  GPR_H08:00000000        GPR_H09:00000000
+0003B0  GPR_H10:00000000        GPR_H11:00000000
+0003B8  GPR_H12:00000000        GPR_H13:00000000
+0003C0  GPR_H14:00000000        GPR_H15:00000000
```

New Function APARs

- **PM04437 – _EDC_PTHREAD_YIELD_MAX**
  - UKxxxxx (R11)
  - Used in conjunction with _EDC_PTHREAD_YIELD
    - _EDC_PTHREAD_YIELD is used to control the amount of time a thread will yield the processor.
    - This value will continually double, up to 32 milliseconds when resource is not available.
  - _EDC_PTHREAD_YIELD_MAX
    - Allows user to set a maximum value for the amount of time to yield up to the 32 millisecond maximum
    - Value set in microseconds (32000 = 32 milliseconds)

New Function APARs

- **PK99010 – CEEPIPI call_sub with writable static**
  - **UK52873 – R9**
  - **UK52874 – R10**
  - **UK52875 – R11**
  - **CEEPIPI call_sub of a subroutine with writable static where C/C++ is not the language of the entry point**
    - **Supports CEEFETCH and CEEPGFD as well**
  - **NOTE: PE – PM27753**
    - **Affects AMODE 24 programs statically linked with C**
    - **C is inherently 31bit, so code change returned function descriptor in 31bit storage that AMODE 24 program could not access.**
    - **Don't need to be calling the C program to be affected.**

What's new in z/OS R12?

- CEEPRMxx OVR/NONOVR support
- Statement of Direction of run-time option ++USERMODs
- BAM XTIOT support
- Heap Storage Reallocation Performance

CEEPRMxx OVR/NONOVR Support

- CEEPRMxx Override/Nonoverride support
  - Existing syntax will be unchanged and fully supported (no migration action)
  - New syntax will match current CEEDOPT usermod syntax
    - ALL31(ON) – existing
    - ALL31=((ON),OVR) – new
  - This includes "NOxxxxx" options
    - NODEBUG – existing
    - DEBUG=((OFF),OVR) – new
      - Suboption is required for these "NO" options

CEEPRMxx OVR/NONOVR Support

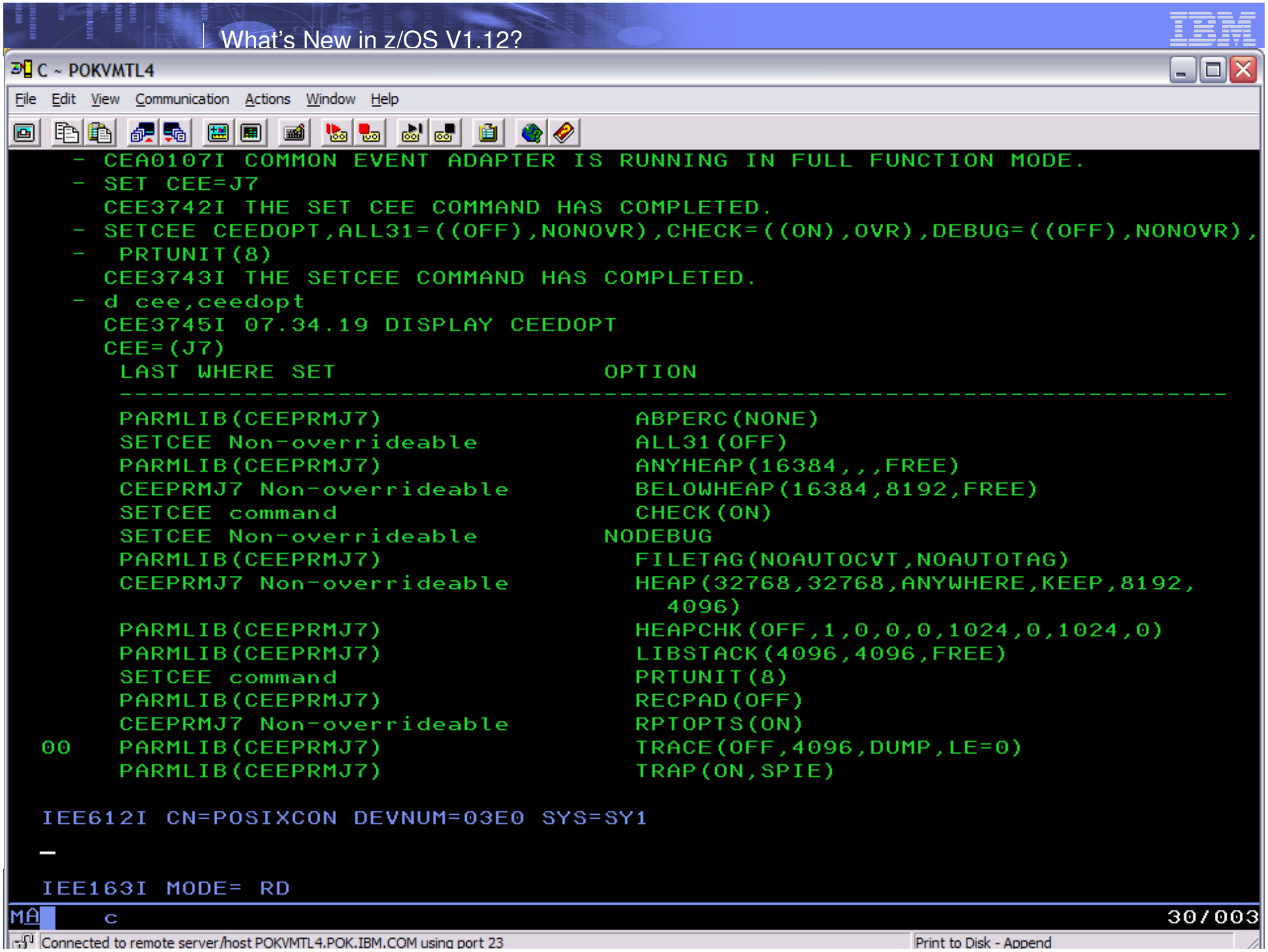
- CEEPRMxx Override/Nonoverride support
  - Will work in SETCEE, SET CEE and syntax checker.
  - D CEE updated to show non-overrideable when appropriate

```
CEE=(A8)
  LAST WHERE SET                  OPTION
  ------------------------------------------------------
  SETCEE Non-overrideable         ALL31(ON)

  CEEPRMA8 Non-overrideable       RPTOPTS(ON)
  CEEPRMA8 Non-overrideable       TRAP(ON,SPIE)
```

CEEPRMxx OVR/NONOVR Support

- CEEPRMxx Override/Nonoverride support
  - SETCEE CEEDOPT,TRAP=((ON),OVR)
    - Acceptable syntax in CEEPRMxx
      - leaving out the 2nd suboption
  - D CEE updated to show only specified suboptions

```
CEE=(A8)
   LAST WHERE SET                        OPTION
   ------------------------------------------------------
   SETCEE Non-overrideable               ALL31(ON)

   CEEPRMA8 Non-overrideable             RPTOPTS(ON)
   SETCEE Non-overrideable               TRAP(ON,)
```


C ~ POKVMTL4
File Edit View Communication Actions Window Help
- CEA0107I COMMON EVENT ADAPTER IS RUNNING IN FULL FUNCTION MODE.
- SET CEE=J7
CEE3742I THE SET CEE COMMAND HAS COMPLETED.
- SETCEE CEEDOPT,ALL31=((OFF),NONOVR),CHECK=((ON),OVR),DEBUG=((OFF),NONOVR),
- PRTUNIT(8)
CEE3743I THE SETCEE COMMAND HAS COMPLETED.
- d cee,ceedopt
CEE3745I 07.34.19 DISPLAY CEEDOPT
CEE=(J7)
LAST WHERE SET OPTION
PARMLIB(CEEPRMJ7) ABPERC(NONE)
SETCEE Non-overrideable ALL31(OFF)
PARMLIB(CEEPRMJ7) ANYHEAP(16384,,,FREE)
CEEPRMJ7 Non-overrideable BELOWHEAP(16384,8192,FREE)
SETCEE command CHECK(ON)
SETCEE Non-overrideable NODEBUG
PARMLIB(CEEPRMJ7) FILETAG(NOAUTOCVT,NOAUTOTAG)
CEEPRMJ7 Non-overrideable HEAP(32768,32768,ANYWHERE,KEEP,8192,
4096)
PARMLIB(CEEPRMJ7) HEAPCHK(OFF,1,0,0,0,1024,0,1024,0)
PARMLIB(CEEPRMJ7) LIBSTACK(4096,4096,FREE)
SETCEE command PRTUNIT(8)
PARMLIB(CEEPRMJ7) RECPAD(OFF)
CEEPRMJ7 Non-overrideable RPTOPTS(ON)
00 PARMLIB(CEEPRMJ7) TRACE(OFF,4096,DUMP,LE=0)
PARMLIB(CEEPRMJ7) TRAP(ON,SPIE)
IEE612I CN=POSIXCON DEVNUM=03E0 SYS=SY1
IEE163I MODE= RD
MA c
30/003
Connected to remote server/host POKVMTL4.POK.IBM.COM using port 23
Print to Disk - Append

Statement of Direction

- IBM plans to remove the capability to change the default Language Environment run-time options settings via SMP/E installable USERMODs. If you wish to change the default Language Environment run-time options for the system, IBM recommends using the CEEPRMxx PARMLIB member.

Statement of Direction

- If you are using the CEEDOPT, CEECOPT or CELQDOPT CSECT to set your installation default run-time options, start using CEEPRMxx NOW.
- If you are using the CEEDOPT, CEECOPT or CELQDOPT CSECT and "cloning" copies of Language Environment modules, see the appendix for CEEROPT Rides Again and Where in the world is CEEDOPT?
- If anyone still believes they need these CSECTs or ++USERMODs please see me!
- Expect this to happen in the release after z/OS V1 R13

BAM XTIOT Support

- Certain Language Environment DDNAMEs now support dynamically allocated with XTIOT, UCB nocapture, or DSAB-above-the-line options specified in the SVC99 parameters (S99TIOEX, S99ACUCB, S99DSABA flags).
  - CEEDUMP DDNAME – supported
  - CEEOPTS DDNAME – NOT supported
  - MSGFILE DDNAME – supported
- C/C++ function fopen() and dynalloc() updated to support the use of new XTIOT options

Heap storage reallocation

- The Language Environment callable service CEECZST (and the C/C++ function realloc()) support a new environment variable
  - _CEE_REALLOC_CONTROL
    - Parameter 1 – Lower bound threshold
      - The number of bytes above which the tolerance percentage (parm 2) will be applied
    - Parameter 2 – Tolerance Percentage
      - The percentage of extra storage to be obtained
      - 0 to 100

Heap storage reallocation

- Example
  - _CEE_REALLOC_CONTROL=100,20
    - First request is for 80 bytes
      - Storage obtained as normal
    - A request to change this storage to 90 bytes
      - Storage obtained as normal
    - A request to change this storage to 100 bytes
      - At or above threshold, percentage is applied
      - Storage obtained is 120 bytes (100 + 100 * 20%)
    - A request to change this storage to 110 bytes
      - No storage need be obtained (we already have 120 bytes)

Heap storage reallocation

- Can be very useful for programs that make many requests to reallocate storage larger than originally requested.
  - Many string manipulation routines make heavy use of storage reallocation.
- If tolerance percentage is 0 or _CEE_REALLOC_CONTROL is not set no change in behavior.

What's new in z/OS R11?

- Assembler Macro Updates
- CICS Additional Floating Point Support
- CELQPIPI service routines update
- Additional diagnostics for HEAPPOOLs

Assembler Macro Updates

- Create CEEGLOB assembler macro similar to IBM Language Environment for z/VSE
- Add support in CEEPPA for the SERVICE keyword option
- Add support in CEEENTRY for the RMODE and AMODE keyword options
- Add support in CEEFETCH to handle both Language Environment and non-Language Environment code and provide support to do an "Language Environment-load" if module previously loaded

Assembler Macro Updates

CEEGLOB global assembler variables:

- **&CEEGPRO (alias &GPRO)** Product number
- **&CEEGVER (alias &GVER)** Product version
- **&CEEGREL (alias &GREL)** Product release
- **&CEEGMOD (alias &GMOD)** Product modification level

- **&CEEGENV (alias &GENV**) OS environment from which the macro has been invoked

Assembler Macro Updates

CEEPPA Service Keyword

- New SERVICE keyword to set the service level string for a routine.
  - `Syntax: SERVICE=service_string`
- The service string length and contents are located following the timestamp and version information.
- This field is not interrogated by Language Environment.
- The SERVICE keyword can only be specified on the first CEEPPA macro in the assembler source, all other instances of the keyword are ignored.
- When the SERVICE keyword is in use, the timestamp is generated automatically, the TSTAMP option is forced to YES even when the user specified TSTAMP=NO.
  - If the TSTAMP option if forced to YES the following severity 4 MNOTE is generated:
    SERVICE PARAMETER SPECIFIED TSTAMP PARAMETER FORCED TO 'YES'

Assembler Macro Updates

CEEPPA Service Keyword

```
192+*         Time Stamp
193+*,Time Stamp = 2009/02/02 15:16:00                          01-CEEPP
194+*,Version 1 Release 1 Modification 0                        01-CEEPP
195+CEETIMES DS    0F                                           01-CEEPP
196+         DC    CL4'2009'          Year                      01-CEEPP
197+         DC    CL2'02'            Month                     01-CEEPP
198+         DC    CL2'02'            Day                       01-CEEPP
199+         DC    CL2'15'            Hours                     01-CEEPP
200+         DC    CL2'16'            Minutes                   01-CEEPP
201+         DC    CL2'00'            Seconds                   01-CEEPP
202+         DC    CL2'1'             Version                   01-CEEPP
203+         DC    CL2'1'             Release                   01-CEEPP
204+         DC    CL2'0'             Modification              01-CEEPP
205+         DC    AL2(6)             Length of Service String  @D2A  01-CEEPP
206+         DC    C'011100'          Service parm              @D2A  01-CEEPP
```

Assembler Macro Updates

Example with CEEGLOB and CEEPPA

```
         GBLC   &GVER,&GREL,&GMOD
         CEEGLOB
ASMTSTRC CEEENTRY PPA=MYPPA,BASE=R11,MAIN=YES
         LA     3,12
         ST     3,RETCODE
         LA     2,8
         LA     3,0
         ST     2,0(,3)
         CEETERM RC=RETCODE,MODIFIER=0
RETCODE  DS     F
R3       EQU    3
R11      EQU    11
         LTORG  ,
* The service level string is set to the concatenation of the CEEGLOB values for
* the Version, Release and Modification Level
MYPPA    CEEPPA SERVICE=&GVER.&GREL.&GMOD
         CEEDSA ,
         CEECAA ,
         CEEOCB ,
         END    ASMTSTRC
```

Assembler Macro Updates

Sample CEEDUMP output

```
Traceback:

DSA    Entry       E  Offset   Statement   Load Mod            Program Unit          Service  Status
1      CEEHDSP     +00004B34                CEEPLPKA            CEEHDSP               HLE7750  Call
2      ASMTSTRC    +0000008A                ASMRC01G            ASMTSTRC              011100   Exception

DSA    DSA Addr    E  Addr     PU Addr     PU Offset  Comp Date  Compile Attributes
1      2159C0B0    0D1BB3E0   0D1BB3E0    +00004B34  20080319   CEL
2      2159C030    0006D000   0006D000    +0000008A  20080512   ASM
```

Assembler Macro Updates

CEEENTRY updated with RMODE and AMODE keyword

- New RMODE and AMODE keywords that will allow for the specification of the modules CSECT RMODE and AMODE settings. The default for both will remain **ANY**.
- Syntax:

  RMODE= <ANY | 24 | 31> the default, if unspecified, is ANY

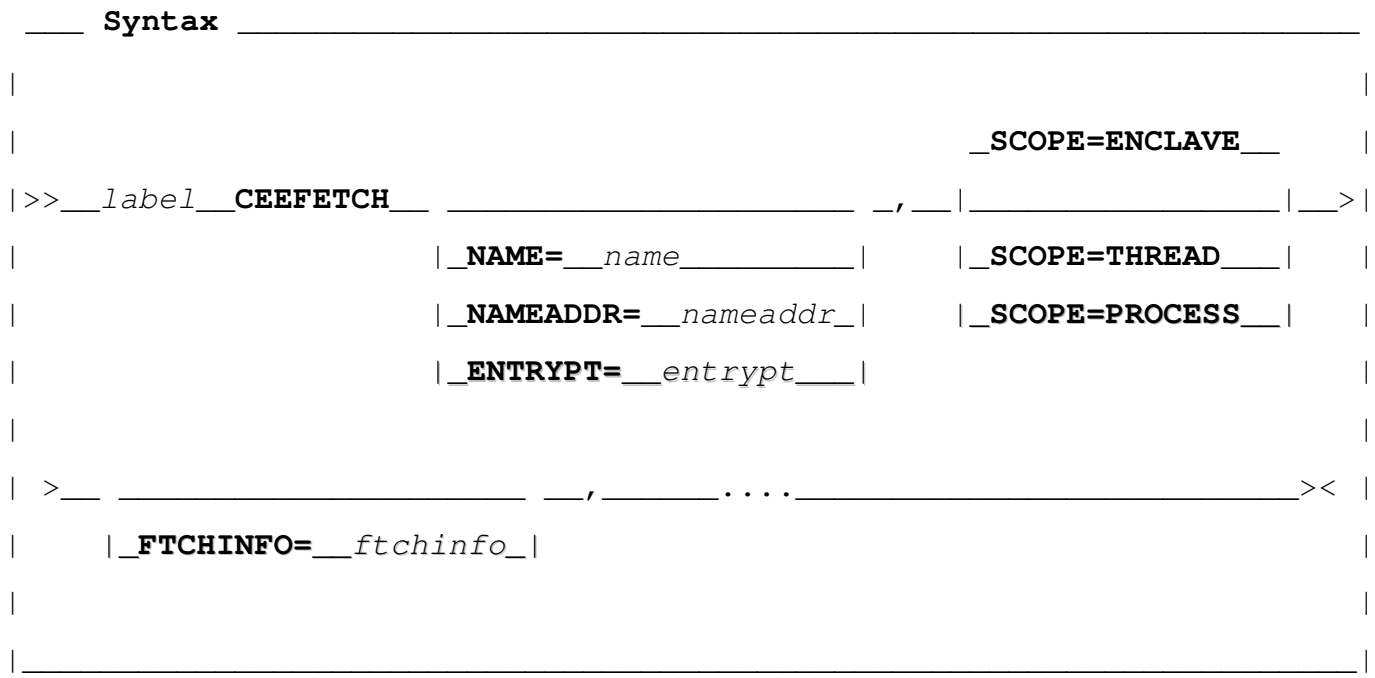
  AMODE= <ANY | 24 | 31 | ANY31> the default, if unspecified, is ANY
- Example:

```
MAIN     CEEENTRY PPA=MAINPPA,…,RMODE=24,AMODE=31
```

Assembler Macro Updates

CEEFETCH Enhancements

- Three new keywords are introduced in CEEFETCH: FTCHINFO, ENTRYPT, and SCOPE=PROCESS



Assembler Macro Updates

CEEFETCH Enhancements

SCOPE=PROCESS

- Indicates that the load is to be scoped to the process level. Modules loaded at the process level are deleted automatically at process termination.
- SCOPE=ENCLAVE remains the default
- SCOPE=THREAD is still supported

Assembler Macro Updates

CEEFETCH Enhancements

FTCHINFO=__*ftchinfo*

- Used in combination with NAME or NAMEADDR to request a *load attempt* on a target module whose characteristics are unknown
- Set to a previously allocated storage area in the form of a register (enclosed in parentheses) or the name of a fullword address variable, that will contain any information discovered about the target module, see CEEFTCH for mapping details
- If the module is identified as a Language Environment conforming AMODE 24 or AMODE 31 subroutine, then processing would be as normal (added to the member list, function pointer obtained, added to the load list table), otherwise only a load of the target will be attempted.

Assembler Macro Updates

CEEFETCH Enhancements

ENTRYPT=__*entrypt*

- Used in combination with FTCHINFO to *obtain information about a previously loaded module and to do any corresponding processing on it as if it was initially loaded by CEEFETCH*
- The *NAME* and *NAMEADDR* keywords are mutually exclusive with ENTRYPT
- If the module is identified as a Language Environment conforming AMODE 24 or AMODE 31 subroutine, then it will be added to the member list, have a function pointer obtained, and added as an entry in to the load list table.
- Set to the entry point for a previously loaded target module stored either in the form of a register (enclosed in parentheses) or the name of a fullword address variable

Assembler Macro Updates

CEEFETCH Enhancements

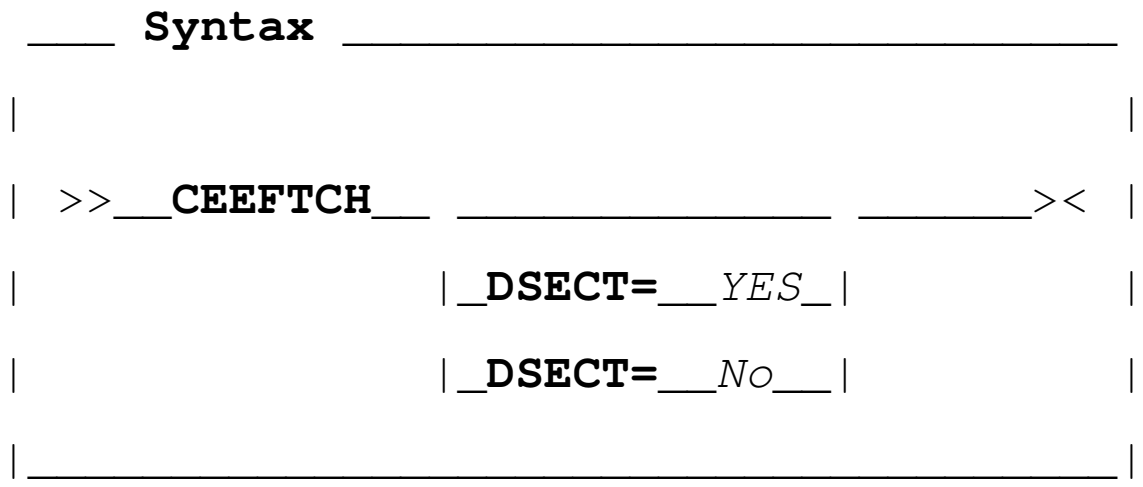
- New messages/feedback codes associated with CEEFETCH

Symbolic Feedback	Severity	Message Number	Message Text
CEE3DV	3	3519	The version specified in the CEEFTCH control block passed to the CEEFETCH macro is not supported.
CEE3QS	1	3932	The system service CSVQUERY failed with return code <return_code> and reason code 0.

Assembler Macro Updates

CEEFTCH

- macro used to generate a mapping for the module information in the FTCHINFO storage area



Assembler Macro Updates

CEEFTCH

- **DSECT=YES**
  - Indicates that a DSECT mapping should be generated.
  - This is the default for the mapping if the DSECT option is not specified.
- **DSECT=NO**
  - Indicates that a data area mapping should be generated.
  - The following tables show the format of the CEEFTCH mapping Version 1 (CEEFTCH_VERSION = 1).
- See APPENDIX for structure details

CICS AFP Support

CICS AFP (Additional Floating Point) Support

- Prior to CICS TS Version 4, Language Environment was unable to fully support Binary Floating Point (BFP) and Decimal Floating Point (DFP)
  - Before this change, Language Environment did not fully support BFP or DFP operations in applications that run in a CICS environment.
    - It was possible to compile XL C/C++ and Enterprise PL/I programs with the AFP(VOLATILE) compiler option and do BFP/DFP operations, as long as the default floating point rounding mode was not altered.
  - In a CICS TS environment, certain BFP and DFP program checks would always result in a CEE3207 message.
    - The same program checks would result in CEE321X, CEE322X, and CEE323X messages in a non-CICS environment.
  - Floating point registers 1,3,5,7, and 8-15, along with the floating point control register (FPC) did not appear in CEEDUMPs or IPCS dumps, when running under CICS TS

CICS AFP Support

CICS AFP (Additional Floating Point) Support

- With this new support, binary and decimal floating point operations are fully supported in the CICS TS Version 4 or later environment.
  - The AFP(VOLATILE) compiler option is no longer required
  - All applicable floating point registers 0-15 and the FPC register appear in dumps after program checks or ABENDs.
  - It is now possible to run many simultaneous programs in a CICS TS region that do binary or decimal floating point operations with non-default rounding modes, with no interference between the applications.

CICS AFP Support

CICS AFP (Additional Floating Point) Support

- Language Environment and CICS TS Version 4 and later will automatically activate the new CICS AFP support when the CICS environment is started
- CEEDUMPs and formatted IPCS dumps will sometimes show additional registers after CICS program checks and ABENDs:
  - Floating point registers 0-15 (before this change only 2, 4, 6, 8 were included)
  - Floating point control register (FPC)
  - High registers (and low registers, as before)
  - Access registers
- Floating point 0C7 program checks are now mapped into the same CEE32xx messages in CICS and non-CICS environments

CELQPIPI Enhancements

CELQPIPI Service Routines

- AMODE 64 Preinitialization (CELQPIPI) previously has supported only 2 service routines:
  - LOAD
  - DELETE
- As of z/OS R11 more service routines will be supported.
  - GETSTORE
  - FREESTORE
  - MSGRTN
- All these service routines are analogous to those routines in AMODE 31 Preinitialization (CEEPIPI).

HEAPPOOLS Diagnostic Enhancements

Enhancements to HEAPPOOLS (and HEAPPOOLS 64) diagnostics

- Format the heap pools structures and storage using IPCS
- Format the heap pools trace with finer granularity
- Limit the heap pools trace to specific pools
- Control the size of the heap pools trace

HEAPPOOLS Diagnostic Enhancements

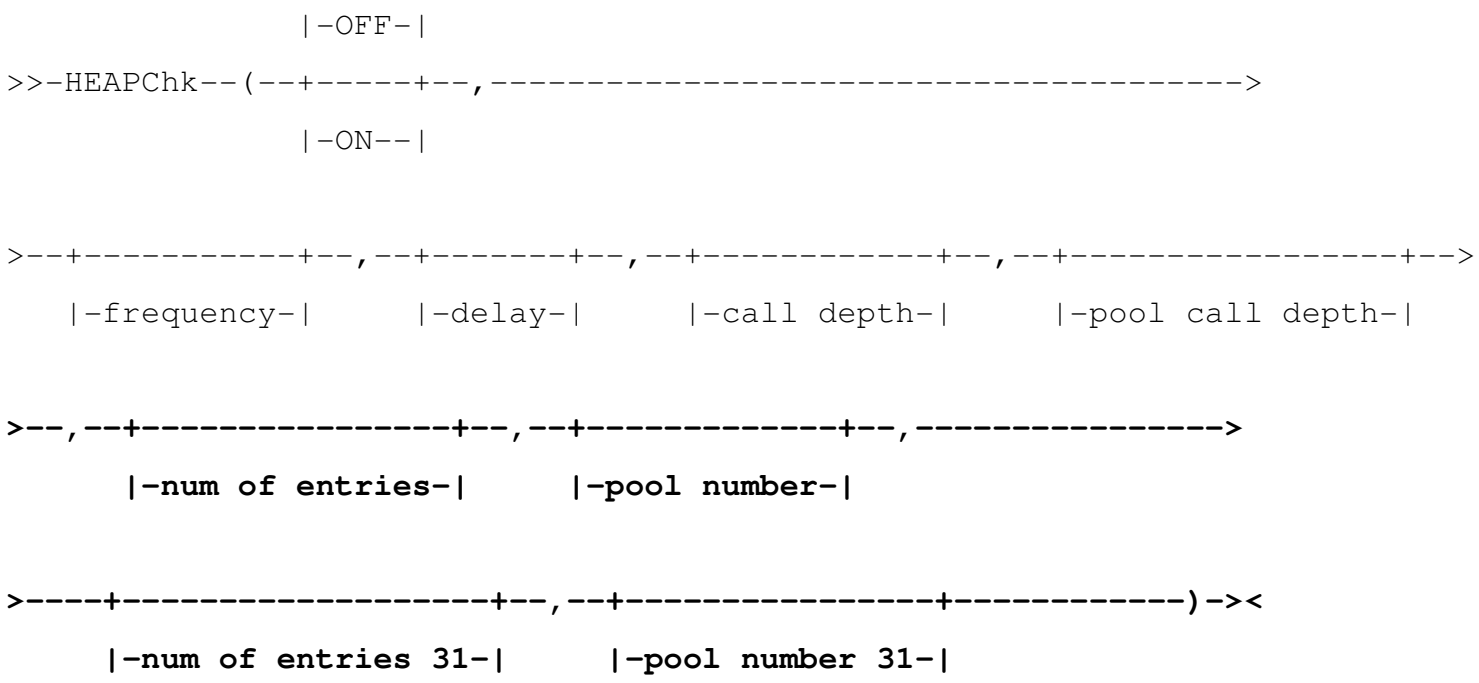
Enhancements to HEAPPOOLS (and HEAPPOOLS 64) diagnostics

- Changes to the HEAPCHK run-time option
  - Four (4) new sub-options are added to the HEAPCHK run-time option
    - Default values provide the same behavior as in prior releases
  - These sub-options control:
    - The number of trace entries per pool (size of the trace)
    - The pool(s) to be traced

HEAPPOOLS Diagnostic Enhancements

Enhancements to HEAPPOOLS (and HEAPPOOLS 64) diagnostics

- Syntax



HEAPPOOLS Diagnostic Enhancements

Enhancements to HEAPPOOLS (and HEAPPOOLS 64) diagnostics

- **Number of Entries**
  - Specifies the number of entries to be recorded in the heap pool trace table for the main user heap in the application. If the heap pool trace table is available and Number of Entries is 0, then the heap pool trace table is not generated.

- **Pool Number**
  - Filter the entries of heap pool trace table recording only those entries of a specific poolid for the main user heap in the application. The value should be a valid pool number (1-12). If heap pool trace table is available and Pool Number is 0 then, the entries of all pools will be traced.

HEAPPOOLS Diagnostic Enhancements

Enhancements to HEAPPOOLS (and HEAPPOOLS 64) diagnostics

- **IPCS – Formatting the heap pools trace**
  - **HPT(*value*) | HPTTCB (*value*) | HPTCELL(*value*) | HPTLOC(*value*)**
    - **HPT - (existing keyword)**
      - If the value is 0 or *, the trace for every heappools poolid is formatted. If the value is a single number (1-12), the trace for the specific heappools poolid is formatted.
    - **HPTTCB**
      - Filters the heappool trace table (if available) printing only those entries for a given TCB address (value).
    - **HPTCELL**
      - Filters the heappool trace table (if available) printing only those entries for a given cell address (value).

HEAPPOOLS Diagnostic Enhancements

Enhancements to HEAPPOOLS (and HEAPPOOLS 64) diagnostics

- **IPCS – Formatting the heap pools trace**
  - **HPT(*value*) | HPTTCB (*value*) | HPTCELL(*value*) | HPTLOC(*value*)**
    - **HPTLOC**
      - Filters the heappool trace table (if available) printing only those entries for a given virtual storage location (value)**.** The valid values are the following**:**
        - 31: Display entries located on virtual storage below the bar
        - 64: Display entries located on virtual storage above the bar
        - ALL: Entries located on virtual storage below / above the bar
  - **NOTE:** Filter options without specifying HPT implies HPT(*).

HEAPPOOLS Diagnostic Enhancements

- IPCS heap pools report
  - Formatted when HEAP or ALL is specified
  - The Heappool report will be very similar to the Heap Report.
  - The report will contain the following information:
    - QPCB
    - QPCB Entry for each pool
    - Addresses
    - Free chain validation
    - Extent validation:
      - Address and size of extent
      - Each free and allocated cell
  - Sample see notes

The End..

Thank you!

Appendix

- CEEFTCH mappings and CEEFETCH example
- What's New in z/OS V1.10?
- Sources for Additional Information

Appendix – CEEFTCH mapping

Offset Dec	Offset Hex	Type	Len	Name(Dim)	Description
0	(0)	Structure	64	CEEFTCH	Start of CEEFETCH
0	(0)	Character	8	CEEFTCH_EYE_CATCHER	Eyecatcher
8	(8)	Unsigned	2	CEEFTCH_VERSION	Version requested
10	(A)	BIT(8)	1	CEEFTCH_FLAGS1	CEEFTCH flags1
10	(A)	BIT(1)	1	CEEFTCH_A24	X'80' target is AMODE 24
10	(A)	BIT(1) POS(2)	1	CEEFTCH_A31	X'40' target is AMODE 31
10	(A)	BIT(1) POS(3)	1	CEEFTCH_A64	X'20' target is AMODE 64
10	(A)	BIT(1) POS(4)	1	CEEFTCH_XPLINK	X'10' target is XPLINK
10	(A)	BIT(1) POS(5)	1	CEEFTCH_LE	X'08' target is Language Environment conforming
10	(A)	BIT(1) POS(6)	1	CEEFTCH_MAIN	X'04' target is MAIN
10	(A)	BIT(1) POS(7)	1	CEEFTCH_SUB	X'02' target is a SUB
10	(A)	BIT(1) POS(8)	1	CEEFTCH_DLL	X'01' target is DLL

Appendix – CEEFTCH mapping

Offset Dec	Offset Hex	Type	Len	Name(Dim)	Description
11	(B)	BIT(8)	1	CEEFTCH_FLAGS2	CEEFTCH flags2
11	(B)	BIT(1)	1	CEEFTCH_SEGMENTED	X'80' target module is divided into multiple initial load segments (deferred load segments, if any, are not counted)
11	(B)	BIT(1) POS(2)	1	CEEFTCH_CICS	X'40' CICS environment
11	(B)	BIT(6) POS(3)	1	*	Available
12	(C)	SIGNED	4	*	Available
16	(10)	ADDRESS	8	CEEFTCH_CEESTART64	Address of 64bit CEESTART
16	(10)	SIGNED	4	*	
20	(14)	ADDRESS	4	CEEFTCH_CEESTART	Address of 31bit CEESTART

Appendix – CEEFTCH mapping

Offset Dec	Offset Hex	Type	Len	Name(Dim)	Description
24	(18)	ADDRESS	8	CEEFTCH_MOD64	Address of 64bit target
24	(18)	SIGNED	4	*	
28	(1C)	ADDRESS	4	CEEFTCH_MOD	Address of 31bit target
32	(20)	SIGNED	8	CEEFTCH_MOD_LEN64	Length of 64bit target
32	(20)	SIGNED	4	*	
36	(24)	SIGNED	4	CEEFTCH_MOD_LEN	Length of 31bit target
40	(28)	ADDRESS	8	CEEFTCH_EP64	Address of 64bit EntryPt
40	(28)	SIGNED	4	*	
44	(2C)	ADDRESS	4	CEEFTCH_EP	Address of 31bit EntryPt
48	(30)	UNSIGNED	8	*	Available
56	(38)	UNSIGNED	8	*	Available

Appendix – ASMFT3E1 example

Example using FTCHINFO to load a module and test the mapping bits to determine characteristics:

```
* ================================================================
* USE NEW FTCHINFO SUPPORT IN CEEFETCH TO ATTEMPT A LOAD
* OF TARGET MODULE 31BIT 'CPPSUBRT'
* ================================================================
ASMFT3E1 CEEENTRY PPA=MYPPA,MAIN=YES,BASE=4,AUTO=WORKSIZE,             X
               ENCLAVE=YES
         USING WORKAREA,13
         LA    2,1
         STH   2,CEEFTCH_VERSION               SET MAP VERSION TO 1
         LA    2,CEEFTCH                       STORE ADDR OF
         ST    2,INFOPT                        CEEFTCH IN INFOPT
         CEEFETCH NAME=CPPSUBRT,                                      X
               TOKEN=TOKEN1,FEEDBACK=FB2,                             X
               MF=(E,LABEL1),FTCHINFO=INFOPT,SCOPE=PROCESS
         CLC   FB2(8),CEE000                   CHECK FEEDBACK CODE
         BE    GOOD_FB
         CALL  CEEMSG,(FB2,DEST,FB3)           DISPLAY FEEDBACK
         CEETERM  RC=16,MODIFIER=0
         B     DONE                            LEAVE IF BAD
GOOD_FB  DS    0H

           BALR   14,15                              31BIT TARGET EXEC
```

Appendix – ASMFT3E1 example

```
* ================================================================
* TEST THE FLAG BITS
* ================================================================
         TM       CEEFTCH_FLAGS1,CEEFTCH_DLL
         JZ       XPLINK_T
         CALL  CEEMOUT,(DLLC,DEST,FB),VL,MF=(E,CALLMOUT)
XPLINK_T EQU      *
         TM       CEEFTCH_FLAGS1,CEEFTCH_XPLINK
         JZ       AMODE_T
         CALL  CEEMOUT,(XPC,DEST,FB),VL,MF=(E,CALLMOUT)
AMODE_T  EQU      *
         TM       CEEFTCH_FLAGS1,CEEFTCH_A24
         JZ       AMODE_3
         CALL  CEEMOUT,(A24C,DEST,FB),VL,MF=(E,CALLMOUT)
AMODE_3  EQU      *
         TM       CEEFTCH_FLAGS1,CEEFTCH_A31
         JZ       AMODE_6
         CALL  CEEMOUT,(A31C,DEST,FB),VL,MF=(E,CALLMOUT)
AMODE_6  EQU      *
         TM       CEEFTCH_FLAGS1,CEEFTCH_A64
         JZ       LE_T
         CALL  CEEMOUT,(A64C,DEST,FB),VL,MF=(E,CALLMOUT)
LE_T     EQU      *
         TM       CEEFTCH_FLAGS1,CEEFTCH_LE
         JZ       SUB_T
         CALL  CEEMOUT,(LEC,DEST,FB),VL,MF=(E,CALLMOUT)
```

Appendix – ASMFT3E1 example

```
SUB_T    EQU      *
         TM       CEEFTCH_FLAGS1,CEEFTCH_SUB
         JZ       MAIN_T
         CALL  CEEMOUT,(SUBC,DEST,FB),VL,MF=(E,CALLMOUT)
MAIN_T   EQU      *
         TM       CEEFTCH_FLAGS1,CEEFTCH_MAIN
         JZ       CICS_T
         CALL  CEEMOUT,(MAINC,DEST,FB),VL,MF=(E,CALLMOUT)
CICS_T   EQU      *
         TM       CEEFTCH_FLAGS2,CEEFTCH_CICS
         JZ       SEG_T
         CALL  CEEMOUT,(CICSC,DEST,FB),VL,MF=(E,CALLMOUT)
SEG_T    EQU      *
         TM       CEEFTCH_FLAGS2,CEEFTCH_SEGMENTED
         B        DONE
         CALL  CEEMOUT,(SEGC,DEST,FB),VL,MF=(E,CALLMOUT)
DONE     DS       0H
*********************************************************************
*        DELETE LOADED ROUTINE                                      *
*********************************************************************
         CEERELES TOKEN=TOKEN1,FEEDBACK=FB2
         CALL  CEEMSG,(FB2,DEST,FB3)                    DISPLAY FB
         CEETERM  RC=0,MODIFIER=0
```

Appendix – ASMFT3E1 example

```
* ================================================================
*        CONSTANTS
* ================================================================
TOKEN1   DS       F
MODNAME  DC       CL8'CPPSUBRT'
FB3      DC       CL12'FEEDBACKCODE'
FB2      DS       CL12'FEEDBACKCODE'
DEST     DC       F'2'            DESTINATION IS THE LE MESSAGE FILE
CEE000   DS       3F'0'           SUCCESS FEEDBACK CODE
*
LEC      DC       Y(LEEND-LESTR)
LESTR    DC       C'I AM LE.'
LEEND    EQU      *
*
A24C     DC       Y(A24END-A24STR)
A24STR   DC       C'I AM AMODE24.'
A24END   EQU      *
*
A31C     DC       Y(A31END-A31STR)
A31STR   DC       C'I AM AMODE31.'
A31END   EQU      *
*
A64C     DC       Y(A64END-A64STR)
A64STR   DC       C'I AM AMODE64.'
A64END   EQU      *
*
```

Appendix – ASMFT3E1 example

```
XPC      DC       Y(XPEND-XPSTR)
XPSTR    DC       C'I AM XPLINK.'
XPEND    EQU      *
*
CICSC    DC       Y(CICSEND-CICSSTR)
CICSSTR  DC       C'I AM IN CICS.'
CICSEND  EQU      *
*
MAINC    DC       Y(MAINEND-MAINSTR)
MAINSTR  DC       C'I AM A MAIN.'
MAINEND  EQU      *
*
SUBC     DC       Y(SUBEND-SUBSTR)
SUBSTR   DC       C'I AM A SUBROUTINE.'
SUBEND   EQU      *
*
DLLC     DC       Y(DLLEND-DLLSTR)
DLLSTR   DC       C'I AM A DLL.'
DLLEND   EQU      *
*
SEGC     DC       Y(SEGEND-SEGSTR)
SEGSTR   DC       C'I AM SEGMENTED.'
SEGEND   EQU      *
```

Appendix – ASMFT3E1 example

```
*
MYPPA    CEEPPA  ,                CONSTANTS DESCRIBING THE CODE BLOCK
* ================================================================
*        THE WORKAREA AND DSA
* ================================================================
WORKAREA DSECT
         ORG   *+CEEDSASZ         LEAVE SPACE FOR THE DSA FIXED PART
FB       DS      3F               SPACE FOR A 12-BYTE FEEDBACK CODE
*
*
CALLMOUT CALL  ,(,,),VL,MF=L      3-ARGUMENT PARAMETER LIST
LABEL1   CEEFETCH MF=L
         CEEFTCH DSECT=NO
INFOPT   DS    A
*
EPPTR    DS    A
*
         DS    0D
WORKSIZE EQU   *-WORKAREA
         CEEDSA  ,                MAPPING OF THE DYNAMIC SAVE AREA
         CEECAA  ,                MAPPING OF THE COMMON ANCHOR AREA
*
*
         END      ASMFT3E1
```

What's new in z/OS R10?

- CEEROPT rides again!
- Where in the world is CEEDOPT?
- Check out that CEEPRMxx member
- Healthy living with LE
- Caught ya – The story of mismatched LE levels
- Pool Party

CEEROPT Rides Again

- CEEROPT for Batch! (and everywhere else)
  - Currently CEEROPT is only processed for CICS and LRR environments
  - Some customers have need to be able to affect run-time options for a large number of applications
    - Without changing installation defaults
    - Without updating JCL
  - Solution CEEROPT for Batch (and more)

CEEROPT Rides Again

- CEEROPT for Batch! (and everywhere else)
  - When enabled
    - Init paths will attempt to load a CEEROPT module from the z/OS search order
      - CELQROPT for AMODE 64
    - If load successful an options merge will take place
      - Between CEEPRMxx and CEEUOPT
    - New CEEWQROP (in SCEESAMP) to assist with CELQROPT creation

CEEROPT Rides Again

- CEEROPT for Batch! (and everywhere else)
  - Controlled by new keywords in CEEPRMxx member
  - CEEROPT keyword
    - COMPAT – just CICS and LRR – default
    - ALL – all initialization paths will attempt to load and use a CEEROPT module
  - CELQROPT keyword (AMODE 64 support – new!)
    - None – Do not attempt to load and use CELQROPT (default)
    - ALL – attempt to load and use CELQROPT

CEEROPT Rides Again

- CEEROPT for Batch! (and everywhere else)
  - Format of the CEEPRMxx member

    CEECOPT(opt1, opt2, ..., optn)

    CEEDOPT(opt1, opt2, ..., optn)

    CELQDOPT(opt1, opt2, ..., optn)

    CEEROPT(ALL|COMPAT)

    CELQROPT(ALL|NONE)
  - Note CEEROPT and CELQROPT are keywords not option groups.
  - CEEROPT and CELQROPT keywords are optional
    - Default is COMPAT and NONE (Same behavior as previous releases)

CEEROPT Rides Again

- CEEROPT for Batch! (and everywhere else)
  - The format of the SETCEE command is:

    SETCEE [CEEDOPT,opt,opt,...]

    [CEECOPT,opt,opt,...]

    [CELQDOPT,opt,opt,...]

    [CEEROPT,ALL|COMPAT]

    [CELQROPT,ALL|NONE]

CEEROPT Rides Again

- CEEROPT for Batch! (and everywhere else)
  - D CEE (just the new stuff…)

```
CEE3745I 11.39.34 DISPLAY CEEROPT
CEE=(MS)
PARMLIB(CEEPRMMS) CEEROPT (ALL)
CEE3745I 11.39.34 DISPLAY CELQROPT
CEE=(MS)
PARMLIB(CEEPRMMS) CELQROPT(NONE)
```

Where in the world is CEEDOPT?

- CEEDOPT is moving!
  - It has been below the line in CEEBINIT, CEEPIPI, CEEBINSS, and others.
  - Moving to CEEPLPKA – above the line
  - Other transparent changes to reduce our below-the-line footprint

Where in the world is CEEDOPT?

- Why do I care that CEEDOPT is moving
  - Most of you don't!
    - If you use CEEPRMxx and don't use the ++USERMOD for CEEDOPT – you don't care
    - If you still use the ++USERMOD for CEEDOPT and only have SMP/e install it – you don't care
    - If you use the ++USERMOD for CEEDOPT and then manually link CEEDOPT into multiple copies of Language Environment modules – you care!!!
      - You have made the teacher mad, you must stay after class!
      - You should be able to use CEEROPT instead
  - See the hopefully entertaining hot topics article – CEEROPT and the Attack of the Clones. (P95) – issue #19
    - http://www-03.ibm.com/systems/z/os/zos/bkserv/hot_topics.html

Check out that CEEPRMxx member

- Syntax Checker for LE PARMLIB (CEEPRMxx)
  - Problem
    - Users of CEEPRMxx could not “test” their changes without activating the changes either by IPL or the SET CEE console command.
  - Solution
    - Provide a syntax checker that can be run in batch or TSO to “pre-test” their changes.

Check out that CEEPRMxx member

- Syntax Checker for LE PARMLIB (CEEPRMxx)
  - Batch
    - The CEEPRMCC program reads and then parses a CEEPRMxx member(s) for syntax errors
      - display a run-time options report if no errors are found.
      - The run-time options report will only display options that are specified inside the CEEPRMxx member(s).

Check out that CEEPRMxx member

- Syntax Checker for LE PARMLIB (CEEPRMxx)
  - Batch
    - Default is to read members from SYS1.PARMLIB
    - CEEPRMCK DD may be used to point to a dataset other than SYS1.PARMLIB

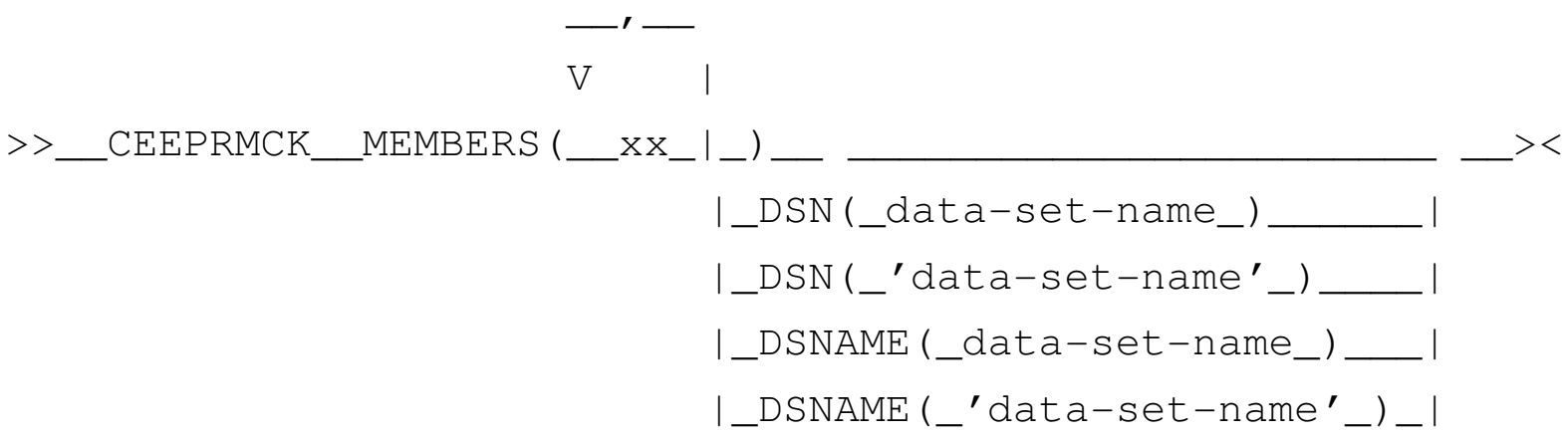
```
//CEEPRMCJ         EXEC PGM=CEEPRMCC,
//                    PARM='CEE=(xx,yy,...,nn)'
//CEEPRMCK    DD DSN=MEENAK.SYSTEM.PARMLIB,DISP=SHR
```

Check out that CEEPRMxx member

- Syntax Checker for LE PARMLIB (CEEPRMxx)
  - TSO
    - The CEEPRMCK CLIST reads and then parses a CEEPRMxx member(s) for syntax errors
      - display a run-time options report if no errors are found.
      - The run-time options report will only display options that are specified inside the CEEPRMxx member(s).

Check out that CEEPRMxx member

- Syntax Checker for LE PARMLIB (CEEPRMxx)
  - TSO
    - The syntax of the CEEPRMCK invocation is as follows:



Check out that CEEPRMxx member

- Syntax Checker for LE PARMLIB (CEEPRMxx)
  - TSO
    - Where:
    - xx
      - The two alphanumeric characters that is the suffix of the CEEPRMxx member(s) to be checked. The MEMBERS keyword parameter must always be specified.
    - data-set-name
      - The data set name that contains the specified CEEPRMxx member. The fully qualified data set name must be enclosed in single quotes if a TSO/E prefix is not desired. The DSN/DSNAME keyword parameter is optional.
      - If both the CEEPRMCK DD is allocated and DSN or DSNAME is specified, then the CEEPRMCK program will use the DD and the DSN/DSNAME will be ignored.
      - If no CEEPRMCK DD is allocated and no DSN or DSNAME is specified, then the CEEPRMCK program will use the default data set SYS1.PARMLIB

Check out that CEEPRMxx member

- Syntax Checker for LE PARMLIB - sample output

```
CEE3762I The Language Environment Parmlib checker has completed.

CEE3745I 11.14.01 Display CEEDOPT
CEE=(ME)

LAST WHERE SET                  OPTION
------------------------------------------------------------------------
PARMLIB(CEEPRMME)                 POSIX(OFF)
PARMLIB(CEEPRMME)                 STORAGE(NONE,NONE,NONE,0)

CEE3745I 11.14.01 Display CEECOPT
CEE=(ME)

LAST WHERE SET                  OPTION
------------------------------------------------------------------------
PARMLIB(CEEPRMME)                 STORAGE(NONE,NONE,20,2048)
```

Healthy Living with LE

- Best practices health check for LE
  - Checker will be shipped active and check for the use of CEEPRMxx to set options group.
  - A successful check will result when CEEPRMxx is used for at least one options group.
  - Name CEE_USING_LE_PARMLIB
  - Rolled back to z/OS R8
    - APAR PK62487 – PTFs will be available by the end of August
      - R8 – UK38468/UK38470
      - R9 – UK38469/UK38471

Caught ya, The story of mismatched LE levels

- There has been many cases of customer problems due to attempting to run Language Environment with modules from 2 different releases.
  - Language Environment will now detect a mismatch during initialization
    - ABEND U4093 Reason code X'F8' (248)
      - Language Environment has detected a mismatch of run-time modules.
      - Check to ensure that run-time modules CEEBINIT, CEEPIPI, CEEBPICI and CEEPLPKA are at the same release level

Caught ya, The story of mismatched LE levels

- Potential causes:
  - Multiple copies of CEEBINIT, CEEPIPI, and CEEBPICI are maintained for unique installation default run-time options or installation default user exits, but they are not fully recreated when a new release is installed.
  - STEPLIBing to a release of Language Environment which is not the same as the z/OS release.
    - If CEEPLPKA is in LPA but CEEBINIT is not, when a spawn() is done, CEEBINIT may not be loaded from the STEPLIB since it has already been loaded for this address space. Either add CEEBINIT to LPA or set the following environment variable: _BPX_SHAREAS=NO

Pool Party

- HEAPPOOLS design change (C/C++ and Enterprise PL/I)
  - There are performance advantages to having multiple pools of the same size "spreading" the accesses around for highly threaded applications.
  - New syntax allows user to specify number of pools for each size
    - HEAPPOOLS=(ON,(8,4),10,(16,3),20,…)
      - 4 pools of size 8 using 10% of init size total
      - 3 pools of size 16 using 20% of init size total
  - Existing HEAPPOOLS syntax continues to be honored
    - HEAPPOOLS=(ON,8,10,16,20,…)
      - Would be the same as HEAPPOOLS=(ON,(8,1),10,(16,1),20…)
  - Storage report updated to assist with tuning of the multiple pools.
  - Also changes involving further alignment on cache lines

Sources for Additional Information

- Language Environment Debugging Guide
- Language Environment Run-Time Messages
- Language Environment Programming Reference
- Language Environment Programming Guide
- Language Environment Programming Guide for 64-bit Virtual Addressing Mode
- Language Environment Customization
- Language Environment Run-Time Application Migration Guide
- Language Environment Writing ILC Applications
- Language Environment Vendor Interfaces
- Language Environment Concepts Guide
- MVS IPCS Commands
- CICS Supplied Transactions