Handling CICS TS for z/VSE Storage Violation Abend Dumps

Mike Poil
michaelalanpoil@gmail.com

Version 4th March 2021

Important Disclaimer

The information contained in this document has not been submitted to any formal test and is distributed on an "as is" basis without any warranty either implied or expressed. The use of this information or the implementation of any of the included techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the operational environment.

Useful documentation

- CICS TS 1.1.0 Problem Determination Guide
- CICS TS 1.1.0 Supplied Transactions
- CICS TS 1.1.0 Trace Entries
- CICS TS 1.1.1 Enhancements Guide
- CICS TS 2.2 Enhancements Guide
- Other Storage Violation analysis PDFs on the Internet, which will be for CICS TS for z/OS, but are still relevant as the underlying causes of the problem are the same.
- Presentations and Tools that I have produced, including the z/VM Rexx FTA EXEC (Full Trace Analyzer) some of whose output is in this presentation, can be found at: http://www.vmworkshop.org/mikepoil/index.shtml
- The content of the above web page will be updated from time-to-time.
- Feel free to contact me if you have any questions.

Agenda

- Introduction
- Possible Causes
- Dumps
- CICS Storage Management Summary
- Storage Check Zones
- CICS Storage Violation Code X'0F0C'
- CICS Storage Violation Trap Code X'0932'
- CICS Storage Violation Code X'0D11'
- CICS Storage Violation Code X'030B'
- Additional Reference Material

Introduction

- After many years in an IBM L3 role, I can say that Storage Overlays are typically the most difficult type of error to diagnose - a CICS Storage Violation Abend is a subset of the types of CICS Storage Overlay that CICS is able to detect, meaning that there are those that CICS isn't able to and won't detect!

- *My personal interpretation of a Storage Overlay is where storage locations other than the expected target of an instruction are modified, hence I would **not** classify placing an invalid value in the correct target field as an overlay*.

- The impact of a CICS Storage Overlay can range from no external visible problem whatsoever to a complete CICS outage - your CICS Applications could be doing it every day and you could be completely unaware of it!

- CICS has built-in diagnostic tools to point you in the right direction, which are the Storage Violation Trap and DFHTRAP, but z/VSE is normally unable to offer any additional diagnostic support in CICS because SDAID does not have anything other than a very simple Storage Overlay detection capability.

Introduction

- The presentation looks at the **initial analysis** of the CICS Storage Violation SM0102 and SM0103 abend dumps, but the **resolution** would normally be a joint effort between Technical Support and somebody in Applications who knows the programs well; the material is relevant to *all* releases of CICS TS on z/VSE.

- At best you will discover that it occurred somewhere between two points of execution in a transaction, but **not** which instruction(s) did the damage.

- Storage Violations are **not** normally the result of a CICS defect, but CICS Support can perform the initial analysis and continue with complex cases.

- SM0102 and SM0103 abends occur when CICS detects that a program's instructions have snagged trip wires added to certain storage allocations acquired by CICS GETMAIN, but overlays elsewhere don't normally result in a Storage Violation abend.

- CICS has other options that can help to further protect from and detect overlays, but they are **not** perfect - they are documented at the end of the presentation.

Introduction

- The Storage Violation abend appears on the console with a "*code*" relating to an Exception "*EXC*" trace entry containing the string "SM *code*", producing a CICS system dump unless it is suppressed; by default, the task is abnormally terminated if not already terminating.

- X'0F0C' occurs during task termination when all remaining task storage is being FREEMAINed when the task is not officially "active", X'0D11' and X'030B' are for an implicit or explicit FREEMAIN, and X'0932' is when the Storage Violation Trap finds a problem.

```
G1 0475 DFHSM0102 IYBFZCCA A storage violation (code X'0F0C') has been detected by module DFHSMAR.
G1 0475 DFHME0116 IYBFZCCA Module:DFHMEME) CICS symptom string for message DFHSM0102 is
        PIDS/564805400 LVLS/430 MS/DFHSM0102 RIDS/DFHSMAR PTFS/zVSE430  PRCS/00000F0C.
G1 0475 DFHDU0201 IYBFZCCA ABOUT TO TAKE SDUMP. DUMPCODE: SM0102   , DUMPID: 1/0002
G1 0475 0S24I AN SDUMP OR SDUMPX MACRO WAS ISSUED
G1 0475 0S29I DUMP STARTED
G1 0475 0S30I DUMP STARTED. MEMBER=DG100898.DUMP IN SUBLIB=SYSDUMP.DYN
G1 0475 1I51I DUMP COMPLETE
G1 0475 DFHDU0202 IYBFZCCA SDUMPX COMPLETE. SDUMPX RETURN CODE X'00'
```

Introduction

- Depending on SIT options and/or prevailing conditions, a Storage Violation abend might result in multiple abends.

- Dumps after the first may be of no use and one you need may be suppressed!

- A Storage Violation might have caused a recent or a subsequent Program Check with an abend like SR0001 due to storage being overlaid, but the Storage Violation *might* follow a Program Check that caused it!

- A Program Check just before the Storage violation can often be fully analyzed using historical abend information kept by the CICS KE (Kernel) Domain.

- A Program Check or 40nn LE abend can occur when LE control block information that is embedded with task storage is destroyed, and the use of the corrupted values has the potential to cause even more damage by LE.

Introduction

- The presentation mentions "CICS-Key" and "User-Key", which is a storage attribute associated with every CICS-managed storage allocation, whether it be task storage or a control block (but see SIT RENTPGM at the end).

- CICS physically separates storage into two classes - CICS and User, and with Storage Protection active (SIT STGPROT=YES) it assigns different Hardware Storage Protection Keys - CICS-Key is the Partition key like X'D' for a Dynamic Partition, but User-Key is always a value of X'9'.

- When a program runs with CICS-Key it can modify almost any area of storage in the Partition and create a lot of damage if it has an error, whereas a program running in User-Key can only modify User-Key storage, which is its own User-Key storage, but it could be *other* task's User-Key storage!

- When Storage Protection is inactive, User-Key = CICS-Key and although storage is still segregated, User-Key programs can overlay CICS-Key storage.

Introduction

- If after reading this material, you find that *your* problem is almost impossible to diagnose - welcome to the real world of CICS problem determination that is our experience in IBM CICS Support!

- However, at least you should now know how to find the information that you need to begin the diagnosis process!

Possible Causes

- A data definition that is bigger than the storage allocation, for example a COMMAREA that is defined as being bigger than was allocated allowing the program to overlay storage past the end.
- A loop uses an index that is bigger than the dimension of the storage array.
- An invalid pointer that has not been initialized, has been corrupted or is no longer valid because the storage has been FREEMAINed and has been reused by another task!
- Assembler code has more potential for errors because it has full control of register usage and pointer maintenance and may fail to do that correctly.
- Non-reentrant (normally Assembler) code can cause other types of obscure overlay problems by changing storage within the program phase.

Dumps

- CICS Support always needs the **unformatted** SYSDUMP library member - for you to look at the problem, format the dump as shown below and keep a backup copy of the dump member.

```
// EXEC INFOANA,SIZE=300K,OS390
   SELECT DUMP MANAGEMENT
     DUMP NAME SYSDUMP.sublib.dumpname
          RETURN
   SELECT DUMP VIEWING
          CALL DFHPD430 DATA AP=3,LD=1,PG=1,TR=3,SM=3,XM=1
          RETURN
   SELECT END
/*
```

- Add KE=3 to look at AP0001 or SR0001 Program Checks that occurred just before the abend - sample output is supplied at the end of the presentation.

- Beware - using IESZNEP instead of the DFHZNEP in PRD1.BASE can cause a control block leak for program IESSVL and PG=1 output can be huge!

Dumps

- AP=3 summarizes *active* transactions and formats their Storage Allocations, but for code X'0F0C' it is not very helpful.

- LD=1 provides program load and entry points and allows you to convert AP 00E1 trace "RET-" (GPR 14) addresses to a program phase + offset value.

- PG=1 provides a program LINK hierarchy so that you can quickly see which program was active at the time of the abend, but not helpful for X'0F0C'.

- SM=3 provides a storage usage summary plus useful SM control blocks.

- TR=3 provides abbreviated and full trace - CICS Support recommends using SIT TRTABSZ=4096 (K) as a **minimum** plus SIT STNTR=1.

- XM=1 can always connect a task number/task id to a transaction id.

Dumps

- I have seen more than 1 million lines produced, and even with a small output file it is useful to know how to find your way around quickly.

- The Domains are printed in ascending order and each one begins with "===AP" or "===TR" (without quotes) etc. and each subsection of the Domain output begins with "==".

- Transaction storage is found by in AP=3 output by searching for ".taskid", e.g., ".78227" with a full stop at the start of the string, and you will see a prefix for the class of storage; each Storage Violation in "active" task storage will have "DFHPD0124" and/or "DFHPD0125" messages next to the allocation.

- PG Domain control blocks are found by searching for "taskid," e.g., "78227,", with a **comma** at the end of the string.

Dumps

- SM Domain control blocks are found by using "00taskid ", e.g., "0078227 ", with a single space at the end of the string, but to be more precise, the search string could be "M00taskid ", "C00taskid ", "B00taskid " or "U00taskid ", and if you were looking for the actual SM task-related control blocks, which are described later, it would be "SCE.x00taskid " and "SCF.x00taskid ".

- CICS-detected "problems" are found in the "===TR" Domain by searching for "*EXC*", and the last one in both the abbreviated *and* the full trace sections should be for the Storage Violation.

- You normally work backwards from the Storage Violation trace entry to see what the task was doing, using the abbreviated trace (the TR=1 part) for speed, and full trace (the TR=2 part) for the detail.

- You may find that a separate execution using CALL DFHPD430 DATA TR=3,TRS=<TASKID=nnnnn>" is easier to read because it only formats trace data from the task id "nnnnn" that was involved in the Storage Violation.

CICS Storage Management Summary

- DSALIM and EDSALIM are fully allocated from Partition GETVIS during CICS initialization, and all storage is marked as "free" at that time; 4 CICS 24-bit "DSA" are dynamically suballocated from free storage in units of 256K "Extents" from DSALIM, plus 4 "EDSA" in 1MB "Extents" from EDSALIM.

- Task-related CICS-Key storage is allocated in the (E)CDSA and User-Key storage in (E)UDSA, being mapped as 4 "Subpools" where each Subpool owns 0 or more 4K pages on a 4K address boundary (the address is divisible by X'1000' with no remainder), although EUDSA uses 64K pages on a 64K address boundary (the address is divisible by X'10000' with no remainder).

- The boundaries provide some degree of storage "isolation" between different task allocations within the same DSA.

- With Storage Protection active, a User-Key program will get a 0C4 Program Check if it tries to modify (E)CDSA or (E)RDSA storage.

CICS Storage Management Summary

- A task acquires storage either from an explicit EXEC CICS GETMAIN request or from a request initiated for it by CICS or LE, and each request is rounded to multiple of 16 bytes on a 16-byte boundary using free space in an existing allocation if possible (first-fit algorithm) or new contiguous Subpool pages.

- New subpool pages may not be contiguous to existing pages, and a loop could modify storage after the end of a task-owned page and overlay another task, however, an overlay in a task's pages normally just affects itself.

- For these requests in the (E)CDSA and (E)UDSA, an 8-byte Storage Check Zone (SCZ) "trip wire" is added to the start *and* end of every allocation and contains a "constant" based on Subpool ID and the task number - a Storage Accounting Area (SAA) is another name for an SCZ.

- If an SCZ is overlaid, CICS will detect the overlay, but normally only at the time when the storage is FREEMAINed - this counts as a Storage Violation and will result in a CICS SM0102 abend.

CICS Storage Management Summary

- The CSFE DEBUG Storage Violation Trap looks at SCZ more frequently (at the time of every EXEC CICS request) and will create an SM0103 abend if it detects an overlay.

- Every transaction requires an amount of storage to be allocated for CICS management purposes, such as TCA control blocks, an EIB and SET storage areas, and CICS will issue *internal* GETMAIN/FREEMAIN requests depending on how the transaction and program has been defined.

- Apart from the task that received the code X'0F0C' abend you will see some control blocks uniquely formatted in AP=3 output, but you will always see the whole allocations, however, the dump itself will always show you both active and residual storage values for *every* task.

- Every explicit GETMAIN without a corresponding FREEMAIN will be formatted in AP=3 as a single allocation.

CICS Storage Management Summary

- Non-LE Assembler programs will have DFHEISTG as a single allocation from the GETMAIN request during program initialization performed by the implicit or explicit DFHEIENT macro, which will be FREEMAINed implicitly by CICS on EXEC CICS RETURN or explicitly by DFHEIRET.

- The address of DFHEISTG can be found in every AP 00E1 ENTRY and EXIT trace, but please remember that DFHEISTG starts with a certain amount of CICS-required storage before the program's storage can be found.

- When LE is used, it becomes slightly more obscure because LE has its own way of managing storage, using CICS GETMAIN and FREEMAIN under the covers, plus SIT RUWAPOOL=YES has an impact on the number of GETMAIN requests that are likely to be required.

- LE needs control blocks to manage the programs and has a habit of placing some control blocks close to task storage, making them easy to overlay and causing LE not to be a happy bunny at times!

CICS Storage Management Summary

- LE uses GETMAIN to acquire at least the minimum amount of a program's LE DSA storage when it starts, e.g. what you see after the COBOL "TGT MEMORY MAP" (TGT, SPEC-REG and WRK-STOR), hence adding an eye-catcher like "WORKING-STORAGE STARTS HERE" can be helpful as there can be several K allocated before WORKING-STORAGE starts.

- LE may make further GETMAIN requests that are the bigger of the required size or what you told LE to use as the secondary allocation size, which is often 4080 so that CICS acquires 4080 + 16 bytes to use a whole 4K page.

- The AP 00E1 trace entries may contain the start address of the LE DSA.

- Whatever, when you look at all the task storage in AP=3 or in the dump itself it can be difficult to determine what is used for what, which is where the insight of an Application person can be vital for solving the problem.

CICS Storage Management Summary

- The (E)CDSA also contain CICS control blocks and CICS-key program phases that do *not* have SCZ and no Storage Violation protection support.
- Task GETMAIN SHARED allocations in the (E)SDSA have no task affinity and User-key program phases that are loaded in these DSAs, hence no SCZ.
- The (E)RDSA only contain SVA-eligible (re-entrant) program phases and should be protected by using SIT RENTPGM=PROTECT.
- RENTPGM=PROTECT use of Storage Protection Key 0 normally stops **all** overlays in the (E)RDSA, causing a 0C4 Program Check if an instruction tries to modify the storage, and the dump PSW will tell you exactly which instruction tried to do it.

Storage Check Zones

- Here are **valid SCZ** for task 00192 in 31-bit CICS-key ("C") ECDSA Task Subpool storage in AP=3 output

```
CICS31.00192 05BAE000 CICS storage above 16MB  ← Note the storage_class.taskid eyecatcher and the SCZ are highlighted
    0000  C3F0F0F0 F0F1F9F2 FFFFFFFF FFFFFFFF  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  *C0000192........................*
    0020  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  06623C68 FFFFFFFF FFFFFFFF FFFFFFFF  *................................*
    0040  FFFFFFFF FFFFFFFF FFFFFFFF 06623458  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  *................................*
    0060  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  *................................*
    0080  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  *................................*
    00A0  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  FFFFFFFF FFFFFFFF C3F0F0F0 F0F1F9F2  *........................C0000192*
```

- 05BAE000 is the SCZ address, but the start address given to the program is 05BAE008; the related GETMAIN trace entry *might* be in the trace data with a "RET-" address that points back to the program phase + offset of the request, but you must use the **last** GETMAIN trace if there are duplicates.

- "B" is 24-bit UDSA, "M" is 24-bit CDSA, "U" is 31-bit EUDSA.

Storage Check Zones

- If the SCZ at the **start** of the storage allocation has been overlaid and that allocation is *not* at the start of a 4K/64K page boundary, it is likely that the owning task caused the overlay.

- Otherwise, see if you can identify which task owns/owned the previous page(s), which you should be able to do using SCZ that are still there - *active* task SCE control blocks in SM=3 output contain the leading SCZ address and the size of the allocation (including SCZ).

- If the SCZ at the **end** of the storage allocation has been overlaid, it is *likely* that the task that owns that page and storage caused the overlay.

- **An overlay that does not damage an SCZ will not be seen by CICS!**

- When an allocation is freed by CICS the content is **not** erased, so what you see before and after an overlaid allocation might be residual data - use SM=3 SCE and SCF control blocks to check the allocation status.

Storage Check Zones

- Eye-catchers and other values in the storage allocation or its size may provide clues about its use if you are unable to find the place in the program where it was allocated, e.g. COBOL has a "3TGT" eye-catcher X'48' bytes (the size of the register save area) after the beginning of the LE DSA start address, and that is followed at a variable offset by the WORKING-STORAGE itself.

- AP=3 shows you the Storage Violations for tasks that were "**active"** at the time, but for abend code X'0F0C' the task was not "**active**", and you won't see the Storage Violation DFHPD012n message relating to the abend.

```
USER24.00192 00A82000 USER storage below 16MB

 0000 00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................* 00A82000

 0020 -    043F LINES SAME AS ABOVE                                                                                 00A82020

** DFHPD0124  Storage violation detected at 00A82000. Leading SAA is invalid.

** DFHPD0125  Storage violation detected at 00A82000. Trailing SAA is invalid.
```

- The above was shown in an X'0F0C' dump but did **not** relate to the Storage Violation that caused the abend - more than one overlay had occurred; the expected SCZ value is "B0000192" as it is USER below storage.

Storage Check Zones

- Terminal-related storage is protected by an **SAA** and it contains the length of the TIOA-8 and an address of the next TIOA or the related TCTTE+4.
- The GETMAIN returns the **actual** TIOA start address.
- I have not seen a dump that was due to an SAA overlay.

```
TIOA.T001  061B7C10  TERMINAL  I/O  AREA

   0000   85000218 061B79F0 00000000 00000000   00000000 00000000 00000000 00000000  *e......0........................*    061B7C10
   0020   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *................................*    061B7C30
   0040 -     01FF LINES SAME AS ABOVE                                                                                          061B7C50
   0200   00000000 00000000 00000000 00000000   00000000 00000000 85000218 061B79F0  *........................e......0*    061B7E10
T001  061B79F0  TERMINAL  I/O  AREA
   0000   85000218 084EAC34 00044040 C3C5D6E3   00000000 00000000 00000000 00000000  *e....+....  CEOT................*    061B79F0
   0020   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *................................*    061B7A10
   0040 -     01FF LINES SAME AS ABOVE                                                                                          061B7A30
   0200   00000000 00000000 00000000 00000000   00000000 00000000 85000218 084EAC34  *........................e....+..*    061B7BF0
```

Storage Violation Abend Code X'0F0C' Analysis Overview

- Find the *EXC* SM 0F0C **full** trace entry with the failing allocation start address, and the formatted allocations will show the overlaid SCZ value(s), storage content that could provide clues about the usage and the overlay value and maybe an SCZ that shows the task number of the allocation owner.

- Go back in the trace data until you find the first "nnnnn" CICS task id (at the time of the abend task "XM" is active) and look in the XM Domain output to find the matching transaction id for task id "nnnnn" - the task id is likely be the same as found in SCZ within the SM 0F0C full trace formatted storage.

- It *might* be possible to find the program phase + X'offset' value at which the GETMAIN occurred and determine what the allocation represents to see how it is used, but the trace entry may no longer be in the formatted trace data and LE-managed programs don't work that way.

- Based on this information it *might* be possible for a programmer to look at the evidence and look at the appropriate program(s) to find the error, but more information may be required.

Storage Violation Abend Code X'0F0C'

- Trace output provides the address of the failing allocation's leading SCZ.
- **In a X'0F0C' dump there could be a *very* long time between the overlay occurring and being detected by CICS!**

```
XM    1 SM 0F0C SMAR  *EXC* Storage_check_failed_at_address 00A81E10 RELEASE_TRANSACTION_STG                      =024602=


SM 0F0C SMAR  *EXC* - Storage_check_failed_at_address - 00A81E10 FUNCTION(RELEASE_TRANSACTION_STG)

   TASK-XM    KE_NUM-0149 TCB-0058C000 RET-9504A982 TIME-15:05:26.8452937189 INTERVAL-00.0000002500    =024602=

     1-0000  00280000 000000D1 00000000 00000000  B0000000 00000000 02000100 00000000  *.......J........................*

       0020  00000000 00000000                                                         *........                        *

     2-0000  00A81E10  ← Address of the start of the allocation                        *....                            *

     3-0000  00000080  ← Allocation size including the SCZ and rounding                *....                            *

     4-0000  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*

       0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*

      Up to 512 bytes of storage is shown beginning at the start of the allocation followed by the same amount of

      storage that includes the end of the allocation - not helpful in this case!

      5-0000  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*

       0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
```

Storage Violation Abend Code X'0F0C

The last two data areas show up to 1K before the start of the allocation and up to 1K after the allocation.

The content of the storage before an overlaid leading SCZ could provide a clue.

However, the 1K of storage before the allocation that has the problem also contains binary zeroes!

The leading SCZ for the next 1K is valid.

```
6-0000  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
  0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
  . . .
  03E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
7-0000  C2F0F0F4 F5F4F5F6 00571000 0001004F  C9C5E2D4 D9E3D740 D9C5C3C5 C9E5C5C4  *B0045456.......|IESMRTP RECEIVED*
  0020  40C1407D C4E2C9C4 C5D9D97D 40D9C5C1  C4C9D5C7 40C9C5E2 E3D9C6D3 4B4040C3  * A 'DSIDERR' READING IESTRFL.  C*
  0040  C8C5C3D2 40E3C8C1 E340C9C5 E2E3D9C6  D340C9E2 40C9D540 C4C6C8C6 C3E34BE2  *HECK THAT IESTRFL IS IN DFHFCT.S*
  . . .
  03C0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
  03E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
```

Storage Violation Abend Code X'0F0C'

- I used our dump browser - on this slide you see 256 bytes *before* the start of the allocation at 00A81E10.

```
A81D10    -100   00000000 00000000 00000000 00000000 | ................ |
A81D20     -F0   00000000 00000000 00000000 00000000 | ................ |
A81D30     -E0   00000000 00000000 00000000 00000000 | ................ |
A81D40     -D0   00000000 00000000 00000000 00000000 | ................ |
A81D50     -C0   00000000 00000000 00000000 00000000 | ................ |
A81D60     -B0   00000000 00000000 00000000 00000000 | ................ |
A81D70     -A0   00000000 00000000 00000000 00000000 | ................ |
A81D80     -90   00000000 00000000 00000000 00000000 | ................ |
A81D90     -80   00000000 00000000 00000000 00000000 | ................ |
A81DA0     -70   00000000 00000000 00000000 00000000 | ................ |
A81DB0     -60   00000000 00000000 00000000 00000000 | ................ |
A81DC0     -50   00000000 00000000 00000000 00000000 | ................ |
A81DD0     -40   00000000 00000000 00000000 00000000 | ................ |
A81DE0     -30   00000000 00000000 00000000 00000000 | ................ |
A81DF0     -20   00000000 00000000 00000000 00000000 | ................ |
A81E00     -10   00000000 00000000 00000000 00000000 | ................ |
```

Storage Violation Abend Code X'0F0C'

```
A81E10      0   00000000 00000000 00000000 00000000 | ................ | No leading SCZ, storage at A81E18
A81E20     10   00000000 00000000 00000000 00000000 | ................ |
A81E30     20   00000000 00000000 00000000 00000000 | ................ |
A81E40     30   00000000 00000000 00000000 00000000 | ................ |
A81E50     40   00000000 00000000 00000000 00000000 | ................ |
A81E60     50   00000000 00000000 00000000 00000000 | ................ |
A81E70     60   00000000 00000000 00000000 00000000 | ................ |
A81E80     70   00000000 00000000 00000000 00000000 | ................ | No trailing SCZ
A81E90     80   C2F0F0F4 F5F4F5F6 00571000 0001004F | B0045456.......| | Next allocation is at +80
A81EA0     90   C9C5E2D4 D9E3D740 D9C5C3C5 C9E5C5C4 | IESMRTP RECEIVED | The leading SCZ is valid
A81EB0     A0   40C1407D C4E2C9C4 C5D9D97D 40D9C5C1 |  A 'DSIDERR' REA |
A81EC0     B0   C4C9D5C7 40C9C5E2 E3D9C6D3 4B4040C3 | DING IESTRFL.  C |
A81ED0     C0   C8C5C3D2 40E3C8C1 E340C9C5 E2E3D9C6 | HECK THAT IESTRF |
A81EE0     D0   D340C9E2 40C9D540 C4C6C8C6 C3E34BE2 | L IS IN DFHFCT.S |
A81EF0     E0   E3D9C6D3 40C9E240 C9D540C4 C6C8C6C3 | TRFL IS IN DFHFC |
A81F00     F0   E34B0000 00000000 C2F0F0F4 F5F4F5F6 | T.......B0045456 |
```

Storage Violation Abend Code X'0F0C'

- If we go back in the abbreviated trace, we can see that the task id before XM gets control to terminate it is 45456.

```
45456 1 AP 0590 APXM  ENTRY RELEASE_XM_CLIENT     NORMAL                                          =024568=

. . .

45456 1 AP 00E7 ERM   EXIT  TASK-CONTROL          ........                                        =024573=

XM    1 SM 0301 SMGF  ENTRY FREEMAIN              14B47944 , 00000050,00AC3080,TCA                =024574=

XM    1 SM 0302 SMGF  EXIT  FREEMAIN/OK                                                           =024575=

XM    1 AP 0591 APXM  EXIT  RELEASE_XM_CLIENT/OK                                                  =024576=

. . .

XM    1 AP 1701 TFIQ  EXIT  SET_TERMINAL_FACILITY/EXCEPTION NO_TERMINAL                           =024599=

XM    1 SM 0401 SMSR  ENTRY INQUIRE_ACCESS        00A8228F,1                                      =024600=

XM    1 SM 0402 SMSR  EXIT  INQUIRE_ACCESS/OK     UDSA,USER                                       =024601=

XM    1 SM 0F0C SMAR  *EXC* Storage_check_failed_at_address 00A81E10 RELEASE_TRANSACTION_STG      =024602=
```

Storage Violation Abend Code X'0F0C'

- In XM=1 output we see that the transaction id for task 45456 is IEGM.

```
Tran Tran  TxnAddr Start Sys  Status   DS       Facility Facility AP       PG       XS       US       RM

id   num   TxdAddr code  Tran          token    type     token    token    token    token    token    token

-------------------------------------------------------------------------------------------------------------

. . .

IEGM 45456 05107100 TT   No   ACT      0D069EA7 None     n/a      00000000 00000000 00000000 00000000 008BA000

           05DBAAC0                                               008BA000 00000000 C0000000 00000000 008BA000
```

Storage Violation Abend Code X'0F0C'

- In SM=3 output there is a summary for task 45456, and later you see the SCE and SCF control blocks that explain how page allocations are mapped in terms of what is in use (SCE) and what is free (SCF).

- The SCEs and SCFs for task 45456 are on the next slide and show how the 8K (2 contiguous 4K pages) of UDSA storage and 64K (1 page) of EUDSA storage is mapped into individual allocations - all still-allocated M0045456 CDSA allocations had already been FREEMAINed before the abend occurred.

```
==SM: Task subpool summary

   Current number of tasks:      18

   SMX Addr Name      Id Loc Acc   Gets  Frees  Elems  Elemstg Pagestg

   . . .

   0502D644 C0045456 03  A   C        0      0      0        0      0K ← ECDSA

            B0045456 02  B   U       17     10      7     4144      8K ← UDSA

            U0045456 04  A   U        2      0      2     4112     64K ← EUDSA
```

Storage Violation Abend Code X'0F0C'

- **SCE** shows **allocated** *UDSA* address and size, **SCF** shows **free** storage.

```
SCE.B0045456 03DD59C8 Storage Element Descriptor

   0000  03ECE830 0405A82C 00A81F10 00000080  14B83A80 00000000

SCE.B0045456 03ECE830 Storage Element Descriptor

   0000  04E0A740 03DD59C8 00A81E90 00000080  14B83A80 0000000

SCE.B0045456 04E0A740 Storage Element Descriptor

   0000  04E0A668 03ECE830 00A81E10 00000080  14B83A80 00000000  ← the failing SCZ address in the 2nd page

SCE.B0045456 04E0A668 Storage Element Descriptor

   0000  03ECE938 04E0A740 00A81D90 00000080  14B83A80 00000000

SCE.B0045456 03ECE938 Storage Element Descriptor

   0000  03DD1128 04E0A668 00A81D10 00000080  14B83A80 00000000

SCE.B0045456 03DD1128 Storage Element Descriptor

   0000  03DD6A88 03ECE938 00A81530 000007E0  14B83A80 00000000 ← 00A81530 is in the 2nd page

SCE.B0045456 03DD6A88 Storage Element Descriptor

   0000  0405A82C 03DD1128 00A80000 000005D0  14B83A80 00000000 ← 00A80000 is the first 4K page start

SCF.B0045456 14B63DB8 Free Storage Descriptor

   0000  0469D0E0 0405A83C 00A805D0 00000F60  14B83A80 00000000 ← 00A805D0 for F60 takes you into the 2nd page

SCF.B0045456 0469D0E0 Free Storage Descriptor

   0000  0405A83C 14B63DB8 00A81F90 00000070  14B83A80 00000000
```

Storage Violation Abend Code X'0F0C'

- SCE shows allocated *EUDSA* address and size, SCF maps free storage.

```
SCE.U0045456 04E6EC68 Storage Element Descriptor
   0000  03E54830 0405A994 056C0FB0 00000060  14F6D000 00000000
SCE.U0045456 03E54830 Storage Element Descriptor
   0000  0405A994 04E6EC68 056C0000 00000FB0  14F6D000 00000000 ← 056C0000 is a 64K page boundary
SCF.U0045456 0471C500 Free Storage Descriptor
   0000  0405A9A4 0405A9A4 056C1010 0000EFF0  14F6D000 00000000    X'10000' or 64K is mapped in total
```

Storage Violation Abend Code X'0F0C'

- The GETMAIN for 00A81E10 with the RET-urn address *after* the EXEC CICS.

```
AP 00E1 EIP ENTRY GETMAIN                                     REQ(0004) FIELD-A(00A83018 .y..) FIELD-B(09000C02 ....)

              TASK-45456 KE_NUM-0149 TCB-0058C000 RET-8095DF58 TIME-15:05:26.8451223439 INTERVAL-00.0000001250    =024203=

. . .

SM 0C01 SMMG  ENTRY - FUNCTION(GETMAIN) GET_LENGTH(67) SUSPEND(YES) INITIAL_IMAGE(00) STORAGE_CLASS(USER24) CALLER(EXEC)

              TASK-45456 KE_NUM-0149 TCB-0058C000 RET-A0CA02AC TIME-15:05:26.8451229689 INTERVAL-00.0000001250    =024207=

                1-0000  00780000 00000011 00000000 00000000  B6780000 00000000 02A80178 C9C5E2D4  *........................y..IESM*

                  0020  D9C54040 0515771A 00000067 00000520  01B83A80 01001201 00000000 00000000  *RE  ...........................*

                  0040  00A73100 85157AE8 00A83080 808C7DC0  056C0008 05158B64 05159DA0 8095E038  *.x..e.:Y.y....'..%..........n..*

                  0060  00AC348C 00000001 00A83220 00040004  00AC0068 00000000                    *.........y.............        *

SM 0C02 SMMG  EXIT  - FUNCTION(GETMAIN) RESPONSE(OK) ADDRESS(00A81E18) ← matching leading SCZ address + 8

              TASK-45456 KE_NUM-0149 TCB-0058C000 RET-A0CA02AC TIME-15:05:26.8451232189 INTERVAL-00.0000002500    =024208=

                1-0000  00780000 00000011 00000000 00000000  B6780000 00000000 02A80178 C9C5E2D4  *........................y..IESM*

                  0020  D9C54040 00A81E18 00000067 00000520  01B83A80 01001201 00000000 00000000  *RE  .y.........................*

                  0040  00A73100 85157AE8 00A83080 808C7DC0  056C0008 05158B64 05159DA0 8095E038  *.x..e.:Y.y....'..%..........n..*

                  0060  00AC348C 00000001 00A83220 00040004  00AC0068 00000000                    *.........y.............        *

. . .

AP 00E1 EIP EXIT GETMAIN OK                                   REQ(00F4) FIELD-A(00000000 ....) FIELD-B(00000C02 ....)

              TASK-45456 KE_NUM-0149 TCB-0058C000 RET-8095DF58 TIME-15:05:26.8451238439 INTERVAL-00.0000001250    =024212=
```

Storage Violation Abend Code X'0F0C'

- Use the return address 0095DF58 (remove the '8' bit) with LD=1 output.

- The offset is 0095DF58 - 0095D2E0 ➔ IESMRTP+C78, remembering that for a CICS "application" program the first X'20' bytes contains the CICS stub module, so IESMRTP+C58 is the actual program listing offset.

```
==LD: PROGRAM STORAGE MAP

PGM NAME ENTRY PT  CSECT   LOAD PT. REL. PTF LVL. LAST COMPILED COPY NO. USERS    LOCN . . .

 . . .

IESOPERR 0095C830 DFHYA430 0095C830 430                               1       0       SDSA

IESMRTP  8095D2E0 DFHYA430 0095D2E0 430                               1       0       SDSA

TSTSSHA1 8095EEB0 DFHYC411 0095EE90 411                               1       0       SDSA
```

- You can use RET- addresses in the trace going back from the SM 0F0C to provide a program+offset traceback to show the code path, but it could be a very slow job! (IBM L2/L3 has the VM FTA EXEC to do this in seconds!)

Storage Violation Abend Code X'0F0C'

- This Storage Violation is in IUI code and without the Storage Violation Trap being active, CICS Support can do nothing, and even z/VSE Support could have a problem diagnosing the cause.

Storage Violation Trap Code X'0932'

- If the SM0102 dump doesn't help, which is common, consider activating the Storage Violation Trap program DFHSMCK to catch the **approximate** point in the transaction where it causes the overlay.

- EXEC CICS calls DFHEIP, producing an AP 00E1 ENTRY trace, followed by SMCK ENTRY and EXIT traces during which the check is made; after the DFHEIP AP 00E1 EXIT trace, SMCK ENTRY and EXIT traces report a second check being made and the application resumes directly after the EXIT trace; if a violation is found, the SMCK EXIT is replaced by abend processing.

- Use CSFE or use SIT CHKSTSK and/or CHKSTRM overrides:

```
CSFE DEBUG,CHKSTSK=ALL            Every task-related storage allocation

CSFE DEBUG,CHKSTSK=CURRENT        The current task's storage allocations

CSFE DEBUG,CHKSTSK=NONE           Disable

CSFE DEBUG,CHKSTRM=CURRENT        The current task's terminal storage

CSFE DEBUG,CHKSTRM=NONE           Disable
```

Storage Violation Trap Code X'0932'

- **CHKSTSK=ALL should always create an SM0103 abend while in the task that overlays the SCZ**, but it can use a lot more cpu time compared to normal because **every** SMCK call checks **every** task's storage allocation, and every EXEC CICS request results in 2 SMCK checks!

- CHKSTSK=CURRENT has a much lower overhead because SMCK only checks the **current** task's storage allocations on every SMCK call, however it only appears to be effective when a task overlays one of its **own** SCZ - if a task overlays **another task's** SCZ, CHKSTSK=CURRENT trace data will report a false positive and suggest that the victim is the cause!

- DFHSMCK will be asked to repair SCZ if SIT STGRCVY=YES is specified, and you will see SMCK FUNCTION(RECOVER_STORAGE) trace entries to show the repair was made.

Storage Violation Trap Code X'0932'

- The trap causes an SM0103 abend, and you can use CSFE to restart the trap if required.

```
G1 0475 DFHSM0103 IYBFZCCA  A storage violation (code X'0932') has been detected by the storage
violation trap. Trap is now inactive.
G1 0475 DFHME0116 IYBFZCCA (Module:DFHMEME) CICS symptom string for message DFHSM0103 is
PIDS/564805400 LVLS/430 MS/DFHSM0103 RIDS/DFHSMCK PTFS/zVSE430 PRCS/00000932.
G1 0475 DFHDU0201 IYBFZCCA ABOUT TO TAKE SDUMP. DUMPCODE: SM0103  , DUMPID: 1/0004
G1 0475 0S24I AN SDUMP OR SDUMPX MACRO WAS ISSUED
G1 0475 0S29I DUMP STARTED
G1 0475 0S30I DUMP STARTED. MEMBER=DG100900.DUMP IN SUBLIB=SYSDUMP.DYN
G1 0475 1I51I DUMP COMPLETE
G1 0475 DFHDU0202 IYBFZCCA SDUMPX COMPLETE. SDUMPX RETURN CODE X'00'
```

Storage Violation Trap Code X'0932'

- Trace entry X'0932'.

```
SM 0932 SMCK  *EXC* - Zone_check_failed - FUNCTION(CHECK_STORAGE) TASK_STORAGE(CURRENT_TASK) TP_STORAGE(NO)

  TASK-00036 KE_NUM-0025 TCB-002F9000 RET-805F6BB2 TIME-10:52:40.7905700334 INTERVAL-00.0000000625    =007824=

     1-0000  00200000 00000010 00000000 00000000  BC000000 00000000 01000100 02010000  *................................*

     2-0000  E4F0F0F0 F0F0F3F6 ← expected SCZ value (the subpool number)             *U0000036                        *

     3-0000  11EBF3B0 ← Address of the start of the allocation                        *..3.                            *

     The first 128 bytes (SCZ OK) and last 16 bytes of the storage allocation (SCZ overlaid)

     4-0000  E4F0F0F0 F0F0F3F6 000801B0 00000000  00000000 00000000 00000000 11EB005C  *U0000036.......................**

       0020  00000000 91DFE8BA 00000000 00000000  11EBF430 11D54120 00000000 11D9D728  *....j.Y..........4..N.......RP.*

       0040  11A18FFF 11A19FFE 11A1AFFD 11EBD920  11EB00D0 11EBF3C8 91DFE4B0 11787680  *.............R.......3Hj.U.....*

       0060  00000000 00000000 00000000 11EBF3C8  00530000 11EB00D0 11EBD920 00000000  *.............3H.........R.....*

     5-0000  20000000 00000000 11EBF550 11D54120 ← trailing SCZ overlaid               *..........5&.N..                *
```

Storage Violation Trap Code X'0932'

- AP=3.

```
USER31.00036 11EBF3B0 USER storage above 16MB

   0000  E4F0F0F0 F0F0F3F6 000801B0 00000000  00000000 00000000 00000000 11EB005C  *U0000036.......................**   11EBF3B0
   0020  00000000 91DFE8BA 00000000 00000000  11EBF430 11D54120 00000000 11D9D728  *....j.Y...........4..N.......RP.*   11EBF3D0
   0040  11A18FFF 11A19FFE 11A1AFFD 11EBD920  11EB00D0 11EBF3C8 91DFE4B0 11787680  *..............R.......3Hj.U.....*   11EBF3F0
   0060  00000000 00000000 00000000 11EBF3C8  00530000 11EB00D0 11EBD920 00000000  *..............3H..........R.....*   11EBF410
   0080  91DFE8F7 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *j.Y7............................*   11EBF430
   00A0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*   11EBF450
   00C0 -    017F LINES SAME AS ABOVE                                                                                11EBF470
   0180  00000000 00000000 00000000 00000000  00000000 00000000 80587DC0 11D54120  *..........................'..N..*   11EBF530
   01A0  08000500 11EBD920 00000000 00000000  20000000 00000000 11EBF550 11D54120  *......R...................5&.N..*   11EBF550
```

```
** DFHPD0125  Storage violation detected at 11EBF3B0. Trailing SAA is invalid.
```

Storage Violation Trap Code X'0932'

- The abbreviated trace shows SMCK entries when the task storage was being checked - any problem will have occurred between the last SMCK EXIT and the SMCK ENTRY just before the abend - what was running during that time?

```
00036 1 AP 00E1 EIP   EXIT  SPOOLWRITE            OK                              00F4,00000000 ....,00005606 ....           =007799=

00036 1 SM 0901 SMCK  ENTRY CHECK_STORAGE         CURRENT_TASK,NO                                                            =007800=

00036 1 SM 0902 SMCK  EXIT  CHECK_STORAGE/OK                                                                                 =007801=

After the SMCK EXIT trace is made, control is passed to the application program and it continues until it performs . . .

00036 1 AP 00E1 EIP   ENTRY LINK                                                  0004,11EBBAC0 ....,0D000E02 ....           =007802=

00036 1 SM 0901 SMCK  ENTRY CHECK_STORAGE         CURRENT_TASK,NO                                                            =007803=

00036 1 SM 0902 SMCK  EXIT  CHECK_STORAGE/OK      NO PROBLEM AT THIS POINT, which was previous SMCK exit                     =007804=

00036 1 AP E160 EXEC  ENTRY LINK                  'DFH£VSAM' AT X'11EBC950','DFHCSD     ................................     =007805=

00036 1 PG 1101 PGLE  ENTRY LINK_EXEC             DFH£VSAM,11EBD920 , 00000074,NO                                            =007806=

00036 1 DD 0301 DDLO  ENTRY LOCATE                11708F80,11D9CFDC,PPT,DFH£VSAM                                             =007807=

00036 1 DD 0302 DDLO  EXIT  LOCATE/OK             D7D7E3C5 , 11FE1930                                                        =007808=

00036 1 LD 0001 LDLD  ENTRY ACQUIRE_PROGRAM       11FE2F10                                                                   =007809=

00036 1 LD 0002 LDLD  EXIT  ACQUIRE_PROGRAM/OK    91DFE490,11DFE490,64A,0,REUSABLE,ECDSA,OLD_COPY                            =007810=

00036 1 AP 1940 APLI  ENTRY START_PROGRAM         DFH£VSAM,CEDF,FULLAPI,EXEC,NO,11D3FC9C,11EBD920 , 00000074,2,NO            =007811=
```

Storage Violation Trap Code X'0932'

```
00036 1 SM 0C01 SMMG  ENTRY GETMAIN               1B0,YES,00,TASK                                                           =007812=
00036 1 SM 0C02 SMMG  EXIT  GETMAIN/OK            11EBF3B8  THE ADDRESS IN WHICH THERE IS A VIOLATION                       =007813=
This is the trace for the DFHTMP Call (locate an FCT entry) in the DFH£VSAM code
00036 1 AP EA00 TMP   ENTRY LOCATE                FCT,DFHCSD                                                                =007814=
00036 1 AP EA01 TMP   EXIT  LOCATE                FCT,DFHCSD,11D54120,NORMAL                                                =007815=
00036 1 AP EA00 TMP   ENTRY UNLOCK                FCT,DFHCSD                                                                =007816=
00036 1 AP EA01 TMP   EXIT  UNLOCK                FCT,DFHCSD,NORMAL                                                         =007817=
The DFHTMP Call completes, the code does more work and the EXEC CICS RETURN is executed
00036 1 AP 00E1 EIP   ENTRY RETURN                                                0004,11EBF3C8 ..3H,0D000E08 ....          =007818=
00036 1 SM 0901 SMCK  ENTRY CHECK_STORAGE         CURRENT_TASK,NO                                                           =007819=
There is no SMCK EXIT, so CICS has found a Storage Violation at the start of DFHEIP processing for the EXEC CICS RETURN, and we see that in
the trace starting with a request that increments a storage violation counter
00036 1 XM 1001 XMIQ  ENTRY SET_TRANSACTION       11707B00 , 0000036C,INCREMENT                                             =007820=
00036 1 XM 1002 XMIQ  EXIT  SET_TRANSACTION/OK                                                                              =007821=
00036 1 AP 1700 TFIQ  ENTRY SET_TERMINAL_FACILITY 11FEC230,YES                                                              =007822=
00036 1 AP 1701 TFIQ  EXIT  SET_TERMINAL_FACILITY/OK                                                                        =007823=
00036 1 SM 0932 SMCK  *EXC* Zone_check_failed     CHECK_STORAGE,CURRENT_TASK,NO   *** NOW THERE IS A PROBLEM ***            =007824=
```

Storage Violation Trap Code X'0932'

- Edited output from my z/VM FTA EXEC using different trace data.

```
Task 00034 STAT EXEC CICS EXIT  SPOOLWRITE           OK       Return DFH0STAT+00021D0A =000037441= API Call Elapsed: 0.0000100625
Task 00034 STAT SMCK CHECK_STORAGE ENTRY =000037442=
Task 00034 STAT SMCK CHECK_STORAGE EXIT  =000037443=
Task 00034 STAT EXEC CICS ENTRY LINK                          Return DFH0STAT+0001C6BC =000037444=
Task 00034 STAT SMCK CHECK_STORAGE ENTRY =000037445=
Task 00034 STAT SMCK CHECK_STORAGE EXIT  =000037446=        All OK at the last SMCK EXIT in the trace
Task 00034 STAT LINK_EXEC PROGRAM(DFH£VSAM)
Task 00034 STAT Dispatched Elapsed: 0.0000039375 Start: =000037436= 12:48:42.2060241411 End: =000037451= 12:48:42.2060280786 TCB: QR...QR
Task 00034 STAT Wait       Elapsed:     0.0000103125 Other wait: CHNGMODE
Task 00034 STAT Dispatched Elapsed: 0 Start: =000037453= 12:48:42.2060383911 End: =000037453= 12:48:42.2060383911 TCB: RO...RO
Task 00034 STAT Wait       Elapsed:     0.0002504004 Other wait: DISPDLAY
Task 00034 STAT LOAD POINT 053F6980 ENTRY POINT 853F6980
Task 00034 STAT Dispatched Elapsed: 0.0074983085 Start: =000037455= 12:48:42.2062887915 End: =000037460= 12:48:42.2137871000 TCB: RO...RO
Task 00034 STAT Wait       Elapsed:     0.0000183750 Other wait: CHNGMODE
Task 00034 STAT EXEC CICS ENTRY RETURN                        Return DFH£VSAM+0000042A =000037479=
Task 00034 STAT SMCK CHECK_STORAGE ENTRY =000037480=      SMCK finds the problem when CICS starts the EXEC CICS RETURN processing
Task 00034 STAT System Abend SM0103
```

Storage Violation Trap Code X'0932'

- **The problem occurred between the beginning of the LINK to DFH£VSAM and its EXEC CICS RETURN.**

- Field A in the AP 00E1 **ENTRY** trace *normally* contains the GPR 13 task DFHEISTG/Working Storage/Stack address, which is 11EBF3C8 in this case, and is the address+8 of the start of the overlaid storage allocation starting at 11EBF3C0.

- In an Assembler program, GPR 13 is always set to the start address of DFHEISTG before the call is made to DFHEIP.

```
00036 1 AP 00E1 EIP   ENTRY RETURN                                                   0004,11EBF3C8 ..3H,0D000E08 .... =007818=

. . .

AP 00E1 EIP ENTRY RETURN                                REQ(0004) FIELD-A(11EBF3C8 ..3H) FIELD-B(0D000E08 ....)

              TASK-00036 KE_NUM-0025 TCB-002F9000 RET-91DFE8BA TIME-10:52:40.7905672209 INTERVAL-00.0000001250    =007818=
```

Storage Violation Trap Code X'0932'

- Using CETR to set EI 1-2 or SIT SYSIPT override STNTREI=(1,2) adds AP E160 ENTRY and AP E161 EXIT trace entries that show more detail.

```
AP 00E1 EIP ENTRY LINK                                  REQ(0004) FIELD-A(11EBBAC0 ....) FIELD-B(0D000E02 ....)

              TASK-00036 KE_NUM-0025 TCB-002F9000 RET-9207A2BC TIME-10:52:40.7905639709 INTERVAL-00.0000001875    =007802=

SM 0901 SMCK  ENTRY - FUNCTION(CHECK_STORAGE) TASK_STORAGE(CURRENT_TASK) TP_STORAGE(NO)

              TASK-00036 KE_NUM-0025 TCB-002F9000 RET-805F6BB2 TIME-10:52:40.7905640334 INTERVAL-00.0000000625    =007803=

                1-0000  00200000 00000010 00000000 00000000  BC000000 00000000 01000000 02010000  *................................*

SM 0902 SMCK  EXIT  - FUNCTION(CHECK_STORAGE) RESPONSE(OK)

              TASK-00036 KE_NUM-0025 TCB-002F9000 RET-805F6BB2 TIME-10:52:40.7905641584 INTERVAL-00.0000001250    =007804=

                1-0000  00200000 00000010 00000000 00000000  BC000000 00000000 01000100 02010000  *................................*

AP E160 EXEC ENTRY LINK PROGRAM('DFH£VSAM' AT X'11EBC950') COMMAREA('DFHCSD     ................................

                                                 ........' AT X'11EBD920') LENGTH(116 AT X'91EBD998') COBOL STMT #(06151)

              TASK-00036 KE_NUM-0025 TCB-002F9000 RET-805F6136 TIME-10:52:40.7905646584 INTERVAL-00.0000005000    =007805=

                1-0000  009E0000 001211EB C9380E02 E0000700  000100F0 F6F1F5F1 0001010C 11EBC950  *........I..........06151......I&*

                  0020  C4C6C85B E5E2C1D4 00020268 11EBD920  C4C6C8C3 E2C44040 40404000 00000000  *DFH£VSAM......R.DFHCSD     .....*

                  0040  00000000 00000000 00000000 00000000  00000000 00000000 00000040 40404040  *...........................     *

                  0060  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                                *

                  0080  40404040 40404040 40404040 00000000  00000000 01030306 91EBD998 0074      *            ............j.Rq..  *

                  . . .
```

Storage Violation Trap Code X'0932'

```
==PG: PTA SUMMARY FOR TRAN NUM : 00036,  PTA ADDRESS : 117FB228 ← note the "taskid,"
LOG-LVL : 2         SYS-LVL : 0         TASK-LLE : 11DEF090  PLCB : 11D9D4E0
=PG: TASK LLE SUMMARY
. . .                                                CA-CURR and CLEN are the
=PG: TASK PLCB SUMMARY                               COMMAREA address and length
PLCB-ADD PROGRAM   LOG-LVL LOAD     ENTRY    LENGTH CA-CURR  CLEN INVK-PRG STG EXIT-NME ENV  PPTE-ADD
11D9D4E0 DFH£VSAM        2 11DFE490 91DFE490 00064A 11EBD920 0074 DFH0STAT                 EXEC 11FE1930 ←active
11D9C860 DFH0STAT        1 1205DC00 9205DC20 025E9F 00000000 0000 CICS                     EXEC 120475D0
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
==XM: TRANSACTION SUMMARY
Tran Tran  TxnAddr Start Sys  Status   DS       Facility Facility AP       PG       XS       US       RM
id   num   TxdAddr code  Tran          token    type     token    token    token    token    token    token
-----------------------------------------------------------------------------------------------------------
STAT 00036 11707B00 T    No   ACT      01800009 Terminal 00000000 11787680 00000000 00000000 1171809F 11797340
           1204A030                                      00000000 00000000 117FB228 00000000 1171A070 11787680
```

Storage Violation Trap Code X'0932'

- So DFH£VSAM has overlaid the trailing SCZ.

- For the Assembler people, here is the problem that was found by the author of the code, which was easy because he put it there to cause the Storage Violation, and real-world problems are more difficult to find!

```
DFHEISTG DSECT
CSAADDR  DS    F
FCTADDR  DS    F
TMPPLIST DFHTM MF=(A,PARMLIST)    DFHTMP parameter list
SAVE     DS    F
         DFHAFCD TYPE=DSECT
. . .
         STM   R0,R15,SAVE        Save all the registers
```

Storage Violation Code X0D11'

- For code X'0D11' the trace shows the start and end of the allocation, and the previous AP 00E1 has the appropriate RET- and R13 DFHEISTG addresses.
- The first byte of the trailing SCZ has been set to X'FF'.

```
00036 1 SM 0D11 SMMF  *EXC* Storage_check_failed_on_freemain_request FREEMAIN,01C00008,EXEC,CICS                =000540=


SM 0D11 SMMF  *EXC* - Storage_check_failed_on_freemain_request - FUNCTION(FREEMAIN) ADDRESS(01C00008) . . .

   TASK-00036 KE_NUM-0023 TCB-002F4000 RET-817962E2 TIME-10:22:11.6346289140 INTERVAL-00.0000003125    =000540=

     1-0000  00780000 00000011 00000000 00000000  B4090000 00000000 046C0148 00000190  *........................%......*

       . . .

     2-0000  01C00000 ← SCZ address                                                *....                            *

     3-0000  E4F0F0F0 F0F0F3F6 00000000 00000000  00000000 00000000 00000000 00000000  *U0000036........................*

       0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*

       0040  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*

       0060  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*

     4-0000  00000000 00000000 FFF0F0F0 F0F0F3F6                                       *.........0000036                *
```

Storage Violation Code X'030B'

- Code X'030B' is like X'0D11'.
- The leading SCZ has been overlaid.

```
34767 1 SM 030B SMGF  *EXC* Storage_check_failed_on_freemain_request FREEMAIN, 01B84248,TASK                  =000859=


SM 030B SMGF  *EXC* - Storage_check_failed_on_freemain_request - FUNCTION(FREEMAIN) ADDRESS(01B84248) STORAGE_CLASS(TASK)
   TASK-34767 KE_NUM-0052 TCB-00481000 RET-814584FE TIME-05:33:47.1293521918 INTERVAL-00.0000002500    =000859=
     1-0000  00400000 0000000E 00000000 00000000  B4080000 00000000 02000100 00000000  *. ..............................*
       0020  00000000 01B84248 00000000 00000000  00000000 00000000 00000200 00000500  *................................*
     2-0000  01B84240 ← SCZ address                                            *...                             *
     3-0000  00000000 00000000 00000000 00000000  00000000 00000000 820AE603 00000000  *........................b.W.....*
       0020  00000000 821E65AE 00000000 C1E2D9C1  021E87AC 00000000 01B83F50 021E64AE  *....b.......ASRA..g........&....*
       0040  00000100 01B848A4 027CD000 01B83F50  01B84258 021E8600 01B84258 007BF080  *.......u.@.....&......f......#0.*
       0060  00000000 00000000 00000000 01B84258  C4C6C8E6 C2C2D3C9 008090D0 01B83F50  *................DFHWBBLI.......&*
     4-0000  00000000 00000000 E4F0F0F3 F4F7F6F7                                      *........U0034767                *
```

Additional Reference Material

- Follows this slide.

Relevant DFHSIT Options

- CMDPROT=YES - checks that User-key EXEC CICS commands don't modify CICS-key storage and abends the task if that would have happened.

- DUMP=YES - dumps like SM0102 can be produced.

- INTTR=ON - internal trace is active at initialisation.

- RENTPGM=PROTECT - protect phases in the (E)RDSA by allacting them in Key 0 storage.

- STGPROT=YES - protect CICS-key storage from User-key programs.

- STGRCVY=NO - don't try to recover from a Storage Violation.
  STGRCVY=YES - *if* the task was "active", DFHSMCK repairs the SCZ *after* the abend and the transaction continues, which is potentially dangerous.

Relevant DFHSIT Options

- STNTR=1 - collect the basic (but *very* useful) CICS trace data entries.
- SYDUMAX=n - allow n CICS system dumps per dump code (e.g. SM0102) to be produced.
- SYSTR=ON - enable standard CICS trace data to be collected.
- TRTABSZ=n - CICS Support recommends a CICS Internal Trace table size of 4096 (K) or higher as the default, but may ask for a larger value for a specific problem; it is allocated in 31-bit GETVIS-ANY storage during CICS initialisation and the size can be increased or decreased dynamically using CETR.
- USERTR=ON - enable EXEC CICS User trace data to be collected.

DFHTRAP

- If there is a persistent overlay that does not create a Storage Violation abend, CICS Support may provide a DFHTRAP diagnostic program that is run when every trace entry is made and which can look for an overlay in CICS control blocks and possibly even in task storage.
- DFHTRAP can use a lot of cpu time to look for an overlay, and in my experience, needed to be iteratively changed to avoid too much cpu time being used.
- Storage overlays can be extremely difficult to diagnose!

SDAID Storage Alteration Trace

- SDAID can be used if the overlay is always at the same storage address and it *will* find the actual instruction that causes the damage.

- The overhead can be **very** high and may not be useable in a Production system.

Edited KE=3 Sample Output

```
......

==KE: KE Domain Error Table Summary

ERR_NUM   ERR_TIME  KE_NUM  ERROR TYPE            ERR_CODE  MODULE    OFFSET
=======   ========  ======  ==========            ========  ======    ======
00000013  08:51:55   0058   TRAN_ABEND_PERCOLATE  ---/ATNI  DFHPCP    000004F2
........
00000039  15:21:58   0020   PROGRAM_CHECK         0C4/AKEA  DFHYI430  00002ED0
0000003A  15:21:58   0020   TRAN_ABEND_PERCOLATE  ---/ASRA  DFHSR1    00000380
0000003B  15:21:58   0020   PROGRAM_CHECK         0C4/AKEA  -noheda-  0000044A
0000003C  15:21:58   0020   TRAN_ABEND_PERCOLATE  ---/ASRA  DFHSR1    00000380
0000003D  15:21:58   0020   PROGRAM_CHECK         0C4/AKEA  -noheda-  000004FC
0000003E  15:21:58   0020   TRAN_ABEND_PERCOLATE  ---/ASRA  DFHSR1    00000380
0000003F  15:21:58   005B   PROGRAM_CHECK         0C1/AKEA  UNKNOWN   UNKNOWN
........
===KE: Kernel Domain Control Blocks

==KE: KE Domain Kernel Storage Report

 KCB 00606000 KERNEL ANCHOR BLOCK

    0000  02106EC4 C6C8D2C5 D2C3C240 40404040  C0038080 C0039B70 C0038480 C004A568  *..>DFHKEKCB     ..........d...v.*    00606000
    0020  C0039910 C0039458 C0038880 C0038C80  C00CA548 40037080 C0038700 C0038F00  *..r...m...h.......v. .....g.....*    00606020
    0040  00000000 00000000 00002328 4000F628  00000005 04456000 40025880 00000000  *............ .6.......-. .......*    00606040
    0060  00000000 E8160000 2EE44000 16980022  00606210 00000000 40025D00 40009080  *....Y....U ..q...-...... .). ...*    00606060
    0080  00000000 7D000000 00000024 0000002C  0000005B 0000A000 00000000 00000000  *....'..............£............*    00606080
    ....
```

Edited KE=3 Sample Output

```
==KE: KE Domain Error Table

=KE: Error Number:  00000039

 KERRD 40026718 KERNEL ERROR DATA

    0000  F0C3F461 C1D2C5C1 010400C4 0000FFFF  C4C6C8C1 D7D3C9F1 2E8D3100 2E6E1280  *0C4/AKEA...D....DFHAPLI1.....>..*    40026718
    0020  00918080 2ED83080 00000039 00000004  079D0004 E0000000 079D2000 B277E8A0  *.j...Q........................Y.*    40026738
    0040  00060004 00000000 B277E8A0 903277E8  2FE66F0C 2FE67170 0083FBDB FFFFD3FD  *..........Y....Y.W?..W...c....L.*    40026758
    0060  0083FBEA FFFFD3FE 2FE049E4 2F874B4F  2FE05988 2FE03988 3277BB28 3277DCC8  *.c....L....U.g.|...h...h.......H*    40026778
    0080  3277BAE4 2FE66DB0 B277E74A 00000000  00000000 00000002 00000000 00000000  *...U.W_...X$....................*    40026798
    00A0  00210003 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    400267B8
    00C0  00000000 00000000 00000000 00000000  079D0004 E0000000 079D2000 B277E8A0  *..............................Y.*    400267D8
    00E0  00060004 00000000 B277E8A0 90000000  2FE66F0C 2FE67170 0083FBDB FFFFD3FD  *..........Y......W?..W...c....L.*    400267F8
    0100  0083FBEA FFFFD3FE 2FE049E4 2F874B4F  2FE05988 2FE03988 3277BB28 3277DCC8  *.c....L....U.g.|...h...h.......H*    40026818
    0120  3277BAE4 2FE66DB0 B277E74A 00000000  00000000 00000002 00000000 00000000  *...U.W_...X$....................*    40026838
    0140  00210003 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    40026858
    0160  00000000 00000000 00000000 00000000  D8E690BD 02268C11 26100000 00000000  *................QW..............*    40026878
    0180  18000000 00000000 3C100000 00027118  34100000 00000000 00000000 00000000  *................................*    40026898
    01A0  00000000 00D80000                                                         *.....Q..                        *    400268B8

   Error Code:  0C4/AKEA    Error Type:  PROGRAM_CHECK           Timestamp:  D8E690BD02268C11

   Reason Code: 00000004

   Date (GMT)   :  27/11/19        Time (GMT)   :  20:21:58.138472
   Date (LOCAL) :  27/11/19        Time (LOCAL) :  15:21:58.138472

   KE_NUM:  0020    KE_TASK:  2ED83080    TCA_ADDR:  00918080    DS_TASK:  2E6E1280

 Error happened in program DFHYI430 at offset 00002ED0

 Registers and PSW.

   PSW:  079D2000 B277E8A0   Instruction Length:  6  Interrupt Code:  04  Exception Address:  00000000

   Execution key at Program Check/Abend: 9                   Branch Event Address: 3277E878
```

Edited KE=3 Sample Output

```
REGISTERS 0-15

REGS 40026768

   0000  2FE66F0C 2FE67170 0083FBDB FFFFD3FD  0083FBEA FFFFD3FE 2FE049E4 2F874B4F  *.W?..W...c....L..c....L....U.g.|*    40026768
   0020  2FE05988 2FE03988 3277BB28 3277DCC8  3277BAE4 2FE66DB0 B277E74A 00000000  *...h...h.......H...U.W_...X$....*    40026788

  Data at PSW: B277E8A0    Module: DFHYI430   Offset: 00002ED0

PSWDATA 3277B9D0

   0000  C4C6C8E8 C9F4F3F0 58F00014 58F0F0B4  58F0F00C 58FF000C 07FF0000 00000000  *DFHYI430.0...00..00.............*    3277B9D0
   0020  47F0F028 00C3C5C5 00000000 00000014  47F0F001 4ACEAC00 3277BA9C 00000000  *.00..CEE.........00.$...........*    3277B9F0
   0040  00000000 00000000 90ECD00C 4110F038  98EFF04C 07FF0000 3277B9F0 3277BAE4  *.............0.q.0<.......0...U*    3277BA10
   0060  327814D8 3277BA50 3277B9F0 3277CF46  32781DE0 3277BAB0 00000000 00000007  *...Q...&...0...................*    3277BA30
   ....
   2DE0  47F0BAFC FA109A7F AD94F811 9A7F9A7F  47F0BAC6 95E89BFB 4780BB1C D2289A82  *.0.....".m8..".".0.FnY......K..b*    3277E7B0
   2E00  A4D99240 9AABD224 9AAC9AAB 92F19A81  47F0BB8E D2019A7F AF6195C1 9A7D4770  *uRk ..K.....k1.a.0..K.."./nA.'..*    3277E7D0
   2E20  BB2E92C2 9A7DFA10 9A7FAD94 F8119A7F  9A7FF911 9A7FAD8D 4720BB5E F871D3C0  *..kB.'...".m8..".."9..".....;8.L.*    3277E7F0
   2E40  9A7F4F20 D3C01A29 D5002BEB 9A7D4770  BB2E47F0 BB8E95C9 9A7E4770 BB7AD22D  *."|.L...N....'.....0..nI.=...:K.*    3277E810
   2E60  9A82A3D3 92409AB0 D21F9AB1 9AB047F0  BB8AD228 9A82A4D9 92409AAB D2249AAC  *.btLk ..K......0..K..buRk ..K...*    3277E830
   2E80  9AAB92F1 9A815820 D28807F2 5820D158  D2019098 241ED201 909A241E D20192CF  *..k1.a..Kh.2..J.K..q..K.....K.k.*    3277E850
   2EA0  820D4130 00014A30 909A4030 909A4820  90981832 4C20A09A 4840909A 18544C40  *b.....$... ......q..<.... ....< *    3277E870
   2EC0  A09A5A40 D1585A20 D158D20B 44142414  D2014420 2420960F 4421D200 44222422  *..! J.!.J.K.....K.....o...K.....*    3277E890
   2EE0  4E30D3C0 F211D3C8 92CF94FC D3C9F921  D3C5D3C8 4780BC14 4B30A086 40309098  *+.L.2.LHk.m.LI9.LELH.......f ..q*    3277E8B0
   2F00  4B50A086 4050909A 47F0BBB6 5820D280  07F24820 90944C20 A09A5A20 D158D20B  *.&.f &...0....K..2...m<...!.J.K.*    3277E8D0
   2F20  24148242 F2122420 90B3960F 242192C1  24225820 D28407F2 5820D158 95C9240E  *..b.2.....o...kA....Kd.2..J.nI..*    3277E8F0
   2F40  4770BC5A 5830D148 5840C02C D2013BD4  401CD507 2008AD43 4770BCFE D2168E88  *...!..J.. ..K..M .N.........K..h*    3277E910
   2F60  AA7E9240 8E9FD234 8EA08E9F D2068DE8  AFD25830 C02CD201 8B203014 D2068DF0  *.=k ..K.....K..Y.K....K.....K..0*    3277E930
   2F80  AF8BD201 8B28301C 41408E88 41508DE8  5040D3C0 5050D3C4 5850D148 41405BC8  *..K...... .h.&.Y& L.&&LD.&J.. £H*    3277E950
   2FA0  41608B20 5040D3C8 5060D3CC 41408DF0  41608E80 5040D3D0 5060D3D4 41408E80  *.-..& LH&-L.. .0.-..& L.&-LM. ..*    3277E970
   2FC0  41608E80 5040D3D8 5060D3DC 41408B28                                      *.-..& LQ&-L.. ..                *    3277E990
```

Edited KE=3 Sample Output

```
Data at Registers

  REG 0   2FE66F0C

31-bit data follows:

REGDATA 2FE66F0C

  -0080  00000000 00000000 00000000 3277BAE4  00000001 2FE67150 2FE66CD8 3277C9E7  *...............U.....W.&.W%Q..IX*    2FE66E8C
  -0060  00000000 B277B9F0 3277BB2C 2FE67010  3277BB14 00000000 2FE03988 00000000  *.......0.....W.............h....*    2FE66EAC


  -0040  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE66ECC
  -0020  00000000 2FE03950 2FE03988 2FE04988  2FE05988 00000000 008610D0 00869008  *.......&...h...h...h.....f...f..*    2FE66EEC

   0000  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE66F0C
   0020 -    00FF LINES SAME AS ABOVE                                                                                    2FE66F2C

24-bit data cannot be accessed

  REG 1   2FE67170

31-bit data follows:

REGDATA 2FE67170

  -0080  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE670F0
  -0060  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE67110
  -0040  00000000 00000000 00000000 00000000  00000000 3277CF46 00000000 80000001  *................................*    2FE67130
  -0020  C0000000 00000000 00000000 00000000  C8C1D5C3 2FE61FD0 2FE61FD0 2FE61FD0  *................HANC.W...W...W..*    2FE67150

   0000  00000000 0011266D 001C4560 2FE03A1A  AFE04560 00000458 00000450 00000000  *......._...-.......-.......&....*    2FE67170
   0020  2FE675E0 00000000 C8C1E340 00000100  00000000 00000000 00000000 00000000  *.W......HAT ....................*    2FE67190
   0040  2FE66DB0 2FE03950 2FE03988 00000000  C8C1D5C3 2FE61FD0 2FE61FD0 2FE61FD0  *.W_....&...h....HANC.W...W...W..*    2FE671B0
   0060  AFE671C0 2FE67F50 00000FF0 00000260  2FE671C0 00000458 00000450 00000000  *.W...W"&...0...-.W.........&....*    2FE671D0
   0080  2FE67640 00000000 C8C1E340 00000100  00000000 00000000 00000000 00000000  *.W. ....HAT ....................*    2FE671F0
   00A0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE67210
   00C0 -    00FF LINES SAME AS ABOVE                                                                                    2FE67230

24-bit data cannot be accessed
```

Edited KE=3 Sample Output

```
REG 2   0083FBDB

31-bit data follows:

REGDATA 0083FBDB

  -0080  00000000 00000000 0001000C 00000000  002F59F4 E0A04000 4C25ED20 1880142F  *...................4.. .<.......*    0083FB5B
  -0060  E8000000 010001DA A0281101 00000000  00404040 40404040 40000000 00000000  *Y................        .......*    0083FB7B
  -0040  00000000 00007DE1 AC000000 00010800  00000000 00000000 00010002 00000000  *......'.........................*    0083FB9B
  -0020  002F59F5 D0A04000 4C25EB80 1880142F  E8000000 010001DA A0281101 00000000  *...5.. .<.......Y...............*    0083FBBB

   0000  00404040 40404040 40000000 00000000  00000000 00007DE1 AC000000 00010800  *.        .............'.........*    0083FBDB
   0020  00000000 00000000 00010009 00000000  002F59F6 C0A04000 4C25F670 1880142F  *...................6.. .<.6.....*    0083FBFB
   0040  E8000000 010001DA A0281101 00000000  00404040 40404040 40000000 00000000  *Y................        .......*    0083FC1B
   0060  00000000 00007DE1 AC000000 00010800  00000000 00000000 00010009 00000000  *......'.........................*    0083FC3B
   0080  002F59F7 B0A04000 4C25EB60 1880142F  E8000000 010001DA A0281101 00000000  *...7.. .<..-....Y...............*    0083FC5B
   00A0  00404040 40404040 40000000 00000000  00000000 00007DE1 AC000000 00010800  *.        .............'.........*    0083FC7B
   00C0  00000000 00000000 00010004 00000000  002F59F8 A0A04000 4C25E910 1880142F  *...................8.. .<.Z.....*    0083FC9B
   00E0  E8000000 010001DA A0281101 00000000  00404040 40404040 40000000 00000000  *Y................        .......*    0083FCBB

24-bit data the same.

  REG 3   FFFFD3FD

31-bit data cannot be accessed **

24-bit data cannot be accessed
```

Edited KE=3 Sample Output

```
  REG 4   0083FBEA

31-bit data follows:

REGDATA 0083FBEA

  -0080  00002F59 F4E0A040 004C25ED 20188014  2FE80000 00010001 DAA02811 01000000  *....4.. .<.......Y..............*    0083FB6A
  -0060  00004040 40404040 40400000 00000000  00000000 0000007D E1AC0000 00000108  *..        .............'........*    0083FB8A
  -0040  00000000 00000000 00000100 02000000  00002F59 F5D0A040 004C25EB 80188014  *....................5.. .<......*    0083FBAA
  -0020  2FE80000 00010001 DAA02811 01000000  00004040 40404040 40404040 00000000  *.Y................        ......*    0083FBCA

   0000  00000000 0000007D E1AC0000 00000108  00000000 00000000 00000100 09000000  *.......'........................*    0083FBEA
   0020  00002F59 F6C0A040 004C25F6 70188014  2FE80000 00010001 DAA02811 01000000  *....6.. .<.6.....Y..............*    0083FC0A
   0040  00004040 40404040 40404040 00000000  00000000 0000007D E1AC0000 00000108  *..        .............'........*    0083FC2A
   0060  00000000 00000000 00000100 09000000  00002F59 F7B0A040 004C25EB 60188014  *....................7.. .<..-...*    0083FC4A
   0080  2FE80000 00010001 DAA02811 01000000  00004040 40404040 40400000 00000000  *.Y................        ......*    0083FC6A
   00A0  00000000 0000007D E1AC0000 00000108  00000000 00000000 00000100 04000000  *.......'........................*    0083FC8A
   00C0  00002F59 F8A0A040 004C25E9 10188014  2FE80000 00010001 DAA02811 01000000  *....8.. .<.Z.....Y..............*    0083FCAA
   00E0  00004040 40404040 40404040 00000000  00000000 0000007D E1AC0000 00000108  *..        .............'........*    0083FCCA

24-bit data the same.

  REG 5   FFFFD3FE

31-bit data cannot be accessed **

24-bit data cannot be accessed
```

Edited KE=3 Sample Output

```
REG 6   2FE049E4

31-bit data follows:

REGDATA 2FE049E4

  -0080  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*   2FE04964
  -0060 -    00FF LINES SAME AS ABOVE                                                                                   2FE04984

24-bit data cannot be accessed

  REG 7   2F874B4F

31-bit data follows:

REGDATA 2F874B4F

  -0080  AC47F0BF A04190D2 24D70790 00900018  19920190 0058F062 3805EF17 FF47F073  *..0....K.P.......k....0.......0.*   2F874ACF
  -0060  9758F062 3C05EF17 FF47F073 9741F000  0413FF47 F0739741 10D22458 90400458  *p.0.......0.p.0.....0.p..K... ..*   2F874AEF
  -0040  90900050 90100058 F079B105 EF12FF47  80BFDA41 F0001047 F0739717 FF582040  *...&....0...........0...0.p.... *   2F874B0F
  -0020  04589020 00582090 14412020 50501020  04582040 04582020 00582020 14412020  *............&&..... ............*   2F874B2F

   0000  50582020 045020D0 F8D207D2 B0797941  00000850 00D0E441 00D2B050 00D0E041  *&....&..8K.K.......&..U..K.&....*   2F874B4F
   0020  00D0E050 00D0C441 00D0E450 00D0C841  00792D50 00D0CC41 2020A850 20D0D041  *...&..D...U&..H....&......y&....*   2F874B6F
   0040  00D0E850 00D0D458 F0901841 10D0C405  EFBFFFD0 E8478070 635820D0 F8179950  *..Y&..M.0.....D.....Y.......8.r&*   2F874B8F
   0060  9020A858 20400458 90200095 05900A47  70739758 20400841 90030050 90200047  *..y.. .....n......p.. .....&....*   2F874BAF
   0080  F0739758 E0400458 90E00041 E0D22458  20901441 20205058 20200450 20E00050  *0.p.. .......K........&....&...&*   2F874BCF
   00A0  90E00441 10D22458 F079AD05 EF12FF47  8070BB41 F0001047 F0739717 FF47F073  *.....K..0...........0...0.p...0.*   2F874BEF
   00C0  975820C2 F458E020 184120E0 50588020  0417EE50 E0D10C58 20400858 202000D5  *p..B4.......&......&.J... .....N*   2F874C0F
   00E0  01200E79 9A477071 C3954B20 12477071  C3581040 0C419000 01509010 00581040  *........Cn......C.. .....&..... *   2F874C2F

24-bit data follows:

REGDATA 00874B4F

  -0080  F2F5F0F1 F1F0F140 40404040 40404040  40404040 40404040 F140000C 40404040  *2501101                 1 ..    *   00874ACF
  -0060  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                                *   00874AEF
  -0040 -    00FF LINES SAME AS ABOVE                                                                                   00874B0F
```

Edited KE=3 Sample Output

```
REG 8   2FE05988

31-bit data follows:

REGDATA 2FE05988

  -0080  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE05908
  -0060  00000000 00000000 00000000 02000000  00004040 00000000 00000000 0000000C  *..................  ............*    2FE05928
  -0040  00000000 00404040 40404040 40404040  40000000 00000000 00000000 00000000  *.....            ...............*    2FE05948
  -0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE05968

   0000  00000000 00000000 02000000 00004040  00000000 00000000 0000000C 00000000  *..............  ................*    2FE05988
   0020  00404040 40404040 40404040 40000000  00000000 00000000 00000000 00000000  *.            ...................*    2FE059A8
   0040  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE059C8
   0060  00000000 02000000 00004040 00000000  00000000 0000000C 00000000 00404040  *..........  .................   *    2FE059E8
   0080  40404040 40404040 40000000 00000000  00000000 00000000 00000000 00000000  *         .......................*    2FE05A08
   00A0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE05A28
   00C0  02000000 00004040 00000000 00000000  0000000C 00000000 00404040 40404040  *......  .................       *    2FE05A48
   00E0  40404040 40000000 00000000 00000000  00000000 00000000 00000000 00000000  *     ...........................*    2FE05A68

24-bit data cannot be accessed
```

Edited KE=3 Sample Output

```
REG 9   2FE03988

31-bit data follows:

REGDATA 2FE03988

  -0080  E4F0F0F7 F8F2F2F6 E4F0F0F7 F8F2F2F6  C8C1D5C3 2FE61FA0 2FE61FA0 00000000  *U0078226U0078226HANC.W...W......*    2FE03908
  -0060  2FE03918 00000000 00002F50 00000000  2FE03918 00002F30 00002F28 00000000  *...........&....................*    2FE03928
  -0040  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE03948
  -0020  E2E8E2D6 E4E34040 00000000 00000000  0E000000 00000000 0F000000 00000000  *SYSOUT  ........................*    2FE03968

   0000  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE03988
   0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE039A8
   0040  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE039C8
   0060  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE039E8
   0080  00000000 00000000 E800C4E2 E9F0F0F0  4040006B 00010000 D3FDD3FE 00000000  *........Y.DSZ000  .,....L.L.....*    2FE03A08
   00A0  00014040 40404040 40404040 40404040  404040F0 F0F1F0F1 F0F14040 F14000F0  *..                 0010101  1 .0*    2FE03A28
   00C0  F0F0F0F0 F0F0F0F0 F0F0F000 00000000  00000000 00000000 00000000 00000000  *00000000000.....................*    2FE03A48
   00E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE03A68

24-bit data cannot be accessed
```

Edited KE=3 Sample Output

```
REG 10  3277BB28

31-bit data follows:

REGDATA 3277BB28

  -0080  FFFFFFBC 3277B9F0 3277BA28 00000008  00000006 3277BA28 32781D60 327818E0  *.......0...................-....*    3277BAA8
  -0060  00000005 00000000 00000000 00000000  00000000 00000000 00000000 40404040  *............................    *    3277BAC8
  -0040  40404040 40404040 40404040 40404040  40404040 40404040 40404000 00000000  *                           .....*    3277BAE8
  -0020  0C000F40 3277BB28 3277CB28 3277CC7C  3277DCC8 3277ECEE 3277FD00 32780D24  *... ...........@...H............*    3277BB08

   0000  3277B9D8 3277CF46 3277CC7C 32781032  3277D278 3277FF7A 3277D5EA 327813C2  *...Q.......@......K....:..N....B*    3277BB28
   0020  3277E644 3277E8E2 3277E908 3277E85C  3277FAC4 3277F812 32780ACE 3277F3E4  *..W...YS..Z...Y*...D..8.......3U*    3277BB48
   0040  3277FA2E 3277F7BE 3277F746 32780748  32780B48 3277FBD4 3277FD06 327808DE  *......7...7............M........*    3277BB68
   0060  327808D8 32780AC8 32780C0A 32780CD2  32780EA0 FFFFFFF1 FFFFFFFF 40000000  *...Q...H.......K.......1.... ...*    3277BB88
   0080  00000000 00000001 001D0015 00140002  000C001E 0010005C 000E000F 002A4B4B  *.......................*........*    3277BBA8
   00A0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    3277BBC8
   00C0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    3277BBE8
   00E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    3277BC08

24-bit data follows:

REGDATA 0077BB28

  -0080  009BC6EA 00000001 009BC6EB 00000001  009BC7EA 00000001 C3E2C140 C1C4C1D6  *..F.......F.......G.....CSA ADAO*    0077BAA8
  -0060  D9404040 01020000 01010202 00000000  00000000 00000100 00000000 00000000  *R   ............................*    0077BAC8
  -0040  F2000490 0077B8A0 00000000 AE912720  AE74E930 80000200 2E9126F6 00000780  *2............j....Z......j.6....*    0077BAE8
  -0020  2E911DA4 00000566 00000000 0077BB68  31EE8000 2FCD454A 32B42030 2F623688  *.j.u...................$.......h*    0077BB08

   0000  2FEC731D 009BC680 80747DC0 00000000  2EDD1B00 00606000 00607230 A2FFFFC4  *......F...'..........--..-..s..D*    0077BB28
   0020  0077BF78 2E912728 F2000338 0077B8A0  A2FFFFC4 2EDDBB50 AE74E930 80000100  *.....j..2.......s..D...&..Z.....*    0077BB48
   0040  009BC780 80684D6E 00685D6C 0080488E  0074994C 00749610 00747648 80685996  *..G...(>..)%......r<..o........o*    0077BB68
   0060  0077BB50 334C9008 0074967C 009BC680  80747DC0 00000000 2EDD1B00 00606000  *...&.<....o@..F...'..........--.*    0077BB88
   0080  00607230 A2FFFFC4 0077BE88 00684F64  0077B8A0 00000008 F900D1C3 000060F8  *.-..s..D...h..|.........9.JC..-8*    0077BBA8
   00A0  F7000000 00000000 00000000 D0010010  0C000413 148F8300 00000000 00000000  *7.....................c.........*    0077BBC8
   00C0  8068593A 32BD7400 00000014 0077BC40  40F0F0F0 F0F0F0F1 00000000 00000000  *...............  0000001........*    0077BBE8
   00E0  00000000 80687346 00000000 00000000  00000000 00000000 806876B2 00686E34  *..............................>.*    0077BC08
```

Edited KE=3 Sample Output

```
REG 11  3277DCC8

31-bit data follows:

REGDATA 3277DCC8

  -0080  4920A086 4720BFE6 06208920 000347F2  BFDE58B0 C03047F0 B474D216 8E88AA95  *...f...W..i....2.......0..K..h.n*    3277DC48
  -0060  92408E9F D2348EA0 8E9FD206 8DE8AFD2  D2068DF0 AF8B4120 8E884140 8DE85020  *k ..K.....K..Y.KK..0.....h. .Y&.*    3277DC68
  -0040  D3C05040 D3C45840 D1484120 4BC84150  8E805020 D3C85050 D3CC4120 8DF05020  *L.& LD. J....H.&..&.LH&&L....0&.*    3277DC88
  -0020  D3D09680 D3D04110 D3C058F0 A0004100  D15C58C0 D08005EF 58C0D0E8 50F0D078  *L.o.L...L..0....J*.........Y&0..*    3277DCA8

   0000  92408267 D24D8268 82675820 D15895E8  269A58B0 C0344770 B040D229 8267A4AF  *k b.K(b.b...J.nY......... K.b.u.*    3277DCC8
   0020  92408291 D2238292 829192F7 82635830  D1485840 C02CD201 3BD4401C 47F0BC40  *k bjK.bkbjk7b...J.. ..K..M ..0. *    3277DCE8
   0040  5830D154 D500301A 8A384770 B0EA956D  81F54780 B0669540 81F54780 B0669500  *..J.N.........n_a5....n a5....n.*    3277DD08
   0060  81F54770 B0EAD501 820DAD88 4780B084  D501820D C0004780 B084D501 820DAAB1  *a5....N.b..h...dN.b......dN.b...*    3277DD28
   0080  4770B0EA D5028227 AD884780 B0ACD502  8227C000 4780B0AC D5028227 AA6C4780  *....N.b..h....N.b.......N.b..%..*    3277DD48
   00A0  B0ACD502 8227AD68 4770B0EA 956D8242  4770B0C2 D50A8243 824258B0 C0384780  *..N.b.......n_b....BN.b.b.......*    3277DD68
   00C0  B2A8D50B 8242C000 58B0C038 4780B2A8  95008242 58B0C034 4770B0EA D50A8243  *.yN.b..........yn.b.........N.b.*    3277DD88
   00E0  824258B0 C0384780 B2A8D500 301A8A3E  58B0C034 4770B126 48402413 49402415  *b........yN.............. ... ..*    3277DDA8

24-bit data follows:

REGDATA 0077DCC8

  -0080  00000000 00000000 00000000 00000000  00000000 00000000 00050B76 00000000  *................................*    0077DC48
  -0060  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    0077DC68
  -0040  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    0077DC88
  -0020  00000000 00000000 00000000 00000000  00000000 3F5023A4 000000AE 2E6FCAAC  *.....................&.u.....?..*    0077DCA8

   0000  00000096 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *...o............................*    0077DCC8
   0020  00000000 00000000 00000000 2EC278C0  0A020000 2EC278C8 0A020000 2EC278D0  *.............B.......B.H.....B..*    0077DCE8
   0040  0A020000 007E7030 0A020000 0077DD28  0077DD30 0077DD38 0077DD40 8077DD48  *.....=..................... ....*    0077DD08
   0060  C0020000 2EC278E0 002B0000 2EC278E0  002C0000 2EC278E0 002D0000 00000000  *.....B.......B.......B..........*    0077DD28
   0080  002E0000 2EC278E0 0077DD58 8077DD68  A0030000 2EC2791C 2EC2791C 00040000  *.....B...............B...B......*    0077DD48
   00A0  000B0000 0077DD74 8077DD84 A0030000  2EC27938 2EC27938 00040000 00110000  *...........d.....B...B..........*    0077DD68
   00C0  0077DD90 8077DDA0 C0030000 2EC27954  2EC27954 00040000 00380000 0077DDB4  *.............B...B..............*    0077DD88
   00E0  0077DDC4 0077DDC8 8077DDCC A0030000  2EC27970 2EC27970 000C0000 00110000  *...D...H.........B...B..........*    0077DDA8
```

Edited KE=3 Sample Output

```
REG 12  3277BAE4

31-bit data follows:

REGDATA 3277BAE4

  -0080  F2F3F0F1 F0F1F0F1 00000000 0000076C  60AE7C0C 80010000 50844B2D 04380042  *23010101.......%-.@.....&d......*    3277BA64
  -0060  01000000 80C08000 00000839 000004CD  20008000 40404040 05000001 32781D60  *....................    .......-*    3277BA84
  -0040  00000000 FFFFFFBC 3277B9F0 3277BA28  00000008 00000006 3277BA28 32781D60  *...........0...................-*    3277BAA4
  -0020  327818E0 00000005 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    3277BAC4

   0000  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404000  *                               .*    3277BAE4
   0020  00000000 0C000F40 3277BB28 3277CB28  3277CC7C 3277DCC8 3277ECEE 3277FD00  *....... ...........@...H........*    3277BB04
   0040  32780D24 3277B9D8 3277CF46 3277CC7C  32781032 3277D278 3277FF7A 3277D5EA  *.......Q.......@......K....:..N.*    3277BB24
   0060  327813C2 3277E644 3277E8E2 3277E908  3277E85C 3277FAC4 3277F812 32780ACE  *...B..W...YS..Z...Y*...D..8.....*    3277BB44
   0080  3277F3E4 3277FA2E 3277F7BE 3277F746  32780748 32780B48 3277FBD4 3277FD06  *..3U......7...7............M....*    3277BB64
   00A0  327808DE 327808D8 32780AC8 32780C0A  32780CD2 32780EA0 FFFFFFF1 FFFFFFFF  *.......Q...H.......K.......1....*    3277BB84
   00C0  40000000 00000000 00000001 001D0015  00140002 000C001E 0010005C 000E000F  * ..........................*....*    3277BBA4
   00E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  * ...............................*    3277BBC4

24-bit data follows:

REGDATA 0077BAE4

  -0080  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00680000  *................................*    0077BA64
  -0060  00000028 00000000 00000000 B5000000  00000000 01000100 00000000 00000000  *................................*    0077BA84
  -0040  D803B190 009BC6EA 00000001 009BC6EB  00000001 009BC7EA 00000001 C3E2C140  *Q.....F.......F.......G.....CSA *    0077BAA4
  -0020  40404040 40404040 01020000 01010202  00000000 00000000 00000100 00000000  *        ........................*    0077BAC4

   0000  00000000 F2000490 0077B8A0 00000000  AE912720 AE74E930 80000200 2E9126F6  *....2............j....Z......j.6*    0077BAE4
   0020  00000780 2E911DA4 00000566 00000000  0077BB68 31EE8000 2FCD454A 32B42030  *.....j.u...................$....*    0077BB04
   0040  2F623688 2FEC731D 009BC680 80747DC0  00000000 2EDD1B00 00606000 00607230  *...h......F...'..........--..-..*    0077BB24
   0060  A2FFFFC4 0077BF78 2E912728 F2000338  0077B8A0 A2FFFFC4 2EDDBB50 AE74E930  *s..D.....j..2.......s..D...&..Z.*    0077BB44
   0080  80000100 009BC780 80684D6E 00685D6C  0080488E 0074994C 00749610 00747648  *......G...(>..)%......r<..o.....*    0077BB64
   00A0  80685996 0077BB50 334C9008 0074967C  009BC680 80747DC0 00000000 2EDD1B00  *...o...&.<....o@..F...'.........*    0077BB84
   00C0  00606000 00607230 A2FFFFC4 0077BE88  00684F64 0077B8A0 00000008 F900D1C3  *.--..-..s..D...h..|.........9.JC*    0077BBA4
   00E0  000060F8 F7000000 00000000 00000000  D0010010 0C000413 148F8300 00000000  *..-87.....................c.....*    0077BBC4
```

Edited KE=3 Sample Output

```
REG 13  2FE66DB0

31-bit data follows:

REGDATA 2FE66DB0

  -0080  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE66D30
  -0060 -   -0001 LINES SAME AS ABOVE                                                                                    2FE66D50

   0000  00108001 2FE65BB8 00000000 B277E74A  00000000 2FE66F0C 2FE67170 3277CB28  *.....W£.......X$.....W?..W......*    2FE66DB0
   0020  008610D0 2FE04560 2FE03A1A 2FE049E4  2F874B4F 2FE05988 2FE03988 3277BB28  *.f.....-.......U.g.|...h...h....*    2FE66DD0
   0040  3277DCC8 2FE64058 F3E3C7E3 2FE65F10  03000000 61030220 2FE66A70 0079A14C  *...H.W .3TGT.W¬...../....W.....<*    2FE66DF0
   0060  2FE671B0 00000000 00002ED5 00000000  00000000 2FE03940 00000000 00000000  *.W.........N........... ........*    2FE66E10
   0080  2FE64058 00000400 00000000 00000000  0000001C 00000001 E2E8E2D6 E4E34040  *.W .....................SYSOUT  *    2FE66E30
   00A0  C9C7E9E2 D9E3C3C4 00000000 00000000  00000000 00000000 00000000 00000000  *IGZSRTCD........................*    2FE66E50
   00C0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    2FE66E70
   00E0  00000000 00000000 3277BAE4 00000001  2FE67150 2FE66CD8 3277C9E7 00000000  *...........U.....W.&.W%Q..IX....*    2FE66E90

24-bit data cannot be accessed
```

Edited KE=3 Sample Output

```
REG 14  B277E74A

31-bit data follows:

REGDATA 3277E74A

  -0080  89200003 47F2BA0A 58B0C034 47F0BB5E  D7129BD8 9BD8D205 9BD8AFEB D2049BE6  *i....2.......0.;P..Q.QK..Q..K..W*    3277E6CA
  -0060  9A785820 C02CD201 90922016 D2108E88  ACCC9240 8E99D23A 8E9A8E99 D2078B78  *......K..k..K..h..k .rK....rK...*    3277E6EA
  -0040  AF9B4140 8E884150 8B785040 D3C05050  D3C44140 9BD84150 90925040 D3C85050  *... .h.&..& L.&&LD. .Q.&.k& LH&&*    3277E70A
  -0020  D3CC4140 9BD85040 D3D09680 D3D04110  D3C058F0 A0004100 D15C58C0 D08005EF  *L.. .Q& L.o.L...L..0....J*......*    3277E72A

   0000  58C0D0E8 50F0D078 5840D158 D5034389  AD804770 BB1C95C1 9BEB4780 BAB8D228  *...Y&0... J.N..i......nA......K.*    3277E74A
   0020  9A82A4D9 92409AAB D2249AAC 9AAB92F1  9A8147F0 BB8E95D5 9BFB4770 BAFCD201  *.buRk ..K.....k1.a.0..nN......K.*    3277E76A
   0040  9A7FAFF9 F9119A7F AD8D4720 BAFCF871  D3C09A7F 4F20D3C0 1A299540 2C1D47D0  *.".99..".......8.L.."|.L...n ....*    3277E78A
   0060  BAEC92E8 9BFB47F0 BAFCFA10 9A7FAD94  F8119A7F 9A7F47F0 BAC695E8 9BFB4780  *..kY...0.....".m8..".".0.FnY....*    3277E7AA
   0080  BB1CD228 9A82A4D9 92409AAB D2249AAC  9AAB92F1 9A8147F0 BB8ED201 9A7FAF61  *..K..buRk ..K.....k1.a.0..K.."./*    3277E7CA
   00A0  95C19A7D 4770BB2E 92C29A7D FA109A7F  AD94F811 9A7F9A7F F9119A7F AD8D4720  *nA.'....kB.'...".m8..".."9..."....*    3277E7EA
   00C0  BB5EF871 D3C09A7F 4F20D3C0 1A29D500  2BEB9A7D 4770BB2E 47F0BB8E 95C99A7E  *.;8.L.."|.L...N....'.....0..nI.=*    3277E80A
   00E0  4770BB7A D22D9A82 A3D39240 9AB0D21F  9AB19AB0 47F0BB8A D2289A82 A4D99240  *...:K..btLk ..K......0..K..buRk *    3277E82A

24-bit data follows:

REGDATA 0077E74A

  -0080  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    0077E6CA
  -0060  00000000 00000000 00000000 00000000  00000000 00000000 00000000 7BC9D5E3  *............................#INT*    0077E6EA
  -0040  C5D94040 00074040 40404040 00000000  00000000 0000B600 00000000 00000400  *ER  ..      ....................*    0077E70A
  -0020  01000000 00000000 000032BA 4CD80100  00000000 0000009A 30470000 00AF0000  *............<Q..................*    0077E72A

   0000  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    0077E74A
   0020 -    009F LINES SAME AS ABOVE                                                                                       0077E76A
   00A0  00000000 00000000 00000000 00000000  00000000 0000E2C5 C7C5D5E3 D9E80000  *......................SEGENTRY..*    0077E7EA
   00C0  00000000 00000077 E8200077 F0000077  E8A00000 00000000 00000000 00000000  *........Y...0...Y...............*    0077E80A
   00E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    0077E82A
```

Edited KE=3 Sample Output

```
REG 15  00000000

31-bit data that can be accessed follows:

REGDATA 00000000

   0000  00000000 00000000 00000000 00000000  00011A10 00004290 071F0000 00015B38  *..............................£.*    00000000
   0020  070D2000 800DEA50 07DD1000 C0653A5E  00000000 00000000 071F0000 00015B38  *.......&.......;..............£.*    00000020
   0040  6009BFF0 0C000000 C0093A78 00000000  00000000 00000000 040C0000 00018446  *-..0..........................d.*    00000040
   0060  040C0000 80017F2E 000C0000 80018802  04080000 80025D3A 040C0000 0001850E  *......".......h.......).......e.*    00000060
   0080  00000420 00001202 0002006B 00060011  80744400 00040000 00000000 00000004  *...........,....................*    00000080
   00A0  00000000 00000000 00000000 00144000  00000000 00000004 00000000 00005B90  *.............. ...............£.*    000000A0
   00C0  18000000 00000000 FBEBFFFB FEFFFF78  007CE000 00000000 D8E690BD 12DE4E11  *.................@......QW....+.*    000000C0
   00E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    000000E0

24-bit data the same.
```

Trademarks

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

BladeCenter*
DB2*
DS6000*
DS8000*
ECKD
FICON*

GDPS*
HiperSockets
HyperSwap
IBM LinuxONE Emperor
IBM LinuxONE Rockhopper
IBM Z*

IBM z13*
IBM z14*
IBM z15*
OMEGAMON*
Performance Toolkit for VM
Power*
PowerVM

PR/SM
RACF*
Storwize*
System Storage*
System x*
System z*

System z9*
System z10*
Tivoli*
zEnterprise*
z/OS*

zSecure
z/VM*
z Systems*

* Registered trademarks of IBM Corporation