IBM GLOBAL SERVICES

E52 / E53

Problem Determination in VSE Part 1 and 2

Tom Grossheider

zSeries® EXPO
FEATURING Z/OS, Z/VM, Z/VSE AND LINUX ON ZSERIES

September 19 - 23, 2005

San Francisco, CA

RETURN TO INDEX

Problem Determination under VSE/ESA
Session E51/E52

VSE/ESA
Technical Conference
San Francisco, CA
Sept 19 – 23, 2005
Tom Grossheider
tgrossh@us.ibm.com

VSE Runs on Hardware

Machines eat hexadecimal for lunch:

It’s all “ones” and “zeros”:

```
0000 = 0        1000 = 8
0001 = 1        1001 = 9
0010 = 2        1010 = 10 (A)
0011 = 3        1011 = 11 (B)
0100 = 4        1100 = 12 (C)
0101 = 5        1101 = 13 (D)
0110 = 6        1110 = 14 (E)
0111 = 7        1111 = 15 (F)
```

Of course, nothing is quite that simple. Hex is always shown as a byte (8 bits):

Hex	Dec
00	00
40	64
80	128
C0	196
FF	255

VSE Runs on Hardware

Extended Binary Coded Decimal Interchange Code (EBCDIC)

4A	$	81	a	C1	A	F0	0
4B	.	82	b	C2	B	F1	1
4C	<	83	c	C3	C	F2	2
4D	(	84	d	C4	D	F3	3
4E	+	85	e	C5	E	F4	4
4F	|	86	f	C6	F	F5	5
50	&	87	g	C7	G	F6	6
5A	!	88	h	C8	H	F7	7
5B	$	89	i	C9	I	F8	8
5C	*	91	j	D1	J	F9	9
5D	)	92	k	D2	K		
5E	;	93	l	D3	L		
5F	(	94	m	D4	M		
60	-	95	n	D5	N		
61	/	96	o	D6	O		
6B	,	97	p	D7	P		
6C	%	98	q	D8	Q		
6D	_	99	r	D9	R		
6E	>	A2	s	E2	S		
6F	?	A3	t	E3	T		
7A	:	A4	u	E4	U		
7B	#	A5	v	E5	V		
7C	@	A6	w	E6	W		
7D	'	A7	x	E7	X		
7E	=	A8	y	E8	Y		
7F	"	A9	z	E9	Z		

VSE Runs on Hardware

Speaker Notes:

Of course Hex, becomes more interesting when you can do basic arithmetic with it. So, for instance:

```
  4        8        C        44        88        CC
 +4       +8       +C       +44       +88       +CC
  8       10       18        88       110       198
```

And, displacement x'5E4' into a program that starts at x'007E41B8' is:

```
007E41B8
    +5E4
________
007E479C
```

If you need to locate the x'4D'th Placeholder, when each PLH is x'2A8' long, and the pool starts at x'02D45018,
… get out your hex calculator.

Rules for doing Hex arithmetic in your head:

1. Translate each digit to its decimal equivalent … or …

2. Remember a few basic shortcuts:
   a. Arithmetic starts on the right-most column, and proceeds column-by-column to the left (just like in the decimal world).
   b. Anything plus F equals that number minus 1 plus 10. So, 7 + F = 16, and F + F = 1E.
   c. Anything plus E equals that number minus 2 plus 10. so, 7 + E = 15, and E + E = 1C
   d. Anything plus D equals that number minus 3 plus 10 … and so forth.

   e. Anything minus F equals that number plus 1, and borrow one from the column to the left. So, 15 – F = 6, and 135 – F = 126.
   f. Anything minus E equals that number plus 2, and borrow one from the column to the left … and so forth.

Exercise One: Program Check

Exercise 1:

1. Recognize VSE/ESA Dump Format.
2. Locate a hexadecimal address in a dump.
3. Identify the module containing a given address, using:
    - Base Registers
    - Branch and Link registers
    - Scanning for an eyecatcher.

Caution: Here comes the boring stuff.

Branch Instructions:

4770415C	Conditional branch:
47F0415C	Unconditional branch
45E0415C	Branch and Link (BAL):
05FE	Branch and Link Register (BALR):

Exercise One: Program Check

Speaker Notes:

Understanding the following machine instructions will be helpful in this exercise:

47C0RDDD Conditional branch: Location derived from adding **DDD** to the contents of base register **R**. This instruction uses the condition code (**C**) set by a previous instruction (compare or test).

e.g. **4770415C =** Branch to Register 4 + x'**15C**', if previous check was "not equal".

4780415C = Branch to above address, if field is zero.

so, if Register 4 contains x'00668ED8', the branch would be to address:

```
00668ED8
    +15C
--------
00669034
```

47F0RDDD Unconditional branch

e.g. **47F0415C =** Branch to Register 4 + x'**15C**'

45B0RDDD Branch and Link (BAL): To **DDD** plus base register **R**. Stores next instruction address in register **B**.

e.g. **45E0415C =** Used for sub-routine calls. Branch to Register 4 + x'**15C**'.
In addition, place the return address in register 14.
So, if Register 4 contains x'00668ED8', as above, and the instruction is located at x'00668918', the next instruction would be fetched from x'00669034' (see above), and Register 14 would contain x'0066891C' (point after the BAL instruction).
The BAL (and BALR) instructions are most commonly used to call subroutines. The subroutine can return to the caller by branching back on the BAL register (in this case, 07FE).

05BR Branch and Link Register (BALR): Target address is in register **R**. Return address is saved into register **B**.

e.g. **05E5 =** Branch to where register 5 is pointing, and, in addition, place the return address in register 14.

An interesting variation on a BALR is if the target register is zero. In this case, no branch takes place, but the next sequential address is still placed into the BAL register. This allows a self-relocating program to load the address of the beginning of the program, and then use that register as a base register.

05C0 = Store the address of the next sequential instruction in register 12, but do not branch. This sets register 12 up as a base register.

You can recognize "branch and link" registers by examining the high-order byte in the register. If the program is executing in 31-bit mode, the bal register will have the x'80' bit turned on in the high-order byte. If executing in 24-bit mode, the high-order two bits contain the instruction length. (b'10000000' or x'80' for 4 byte instruction; b'01000000' or x'40' for a two byte instruction).

Exercise One: Program Check

Problem Description:

Getting program check interruption in some COBOL programs. Message 0S03I with interruption code 05 using ISAM interface (IIP).

Console log excerpt:

```
0S03I PROGRAM CHECK INTERRUPTION - HEX LOCATION 001C3E7A -INTERRUPTION CODE 05-
      ADDRESSING EXCEPTION
0S00I JOB CSFLDUPD CANCELED
0S07I PROBLEM PROGRAM  PSW = 071D0000 801C3E7C
0S30I DUMP STARTED. MEMBER=DBG00018.DUMP IN SUBLIB=SYSDUMP.BG
1I49I DUMP LIBRARY FULL
```

Exercise One: Program Check

Speaker Notes:

1. msg0s03i indicates the error occurred at hex location x'001C3E7A'.
2. Interruption code 05 indicates an addressing exception. Once the target address is resolved (contents of base register added to the displacement, the resultant address was invalid (beyond the high address of the partition).
3. Other program interrupt codes are (See *ESA/390 Principles of Operations*)

1	Operation Exception: Data being executed is not a valid instruction. Either an overlay, or a bad branch, which is also probably caused by an overlay.
2	Privileged Operation Exception: Application program is attempting to execute an instruction which is reserved for supervisor state. Probably the same cause as code 1.
3	Execute Exception: Object of an execute instruction is not allowed to be another execute.
4	Protection Exception: Application program is attempting to change storage which is reserved for the supervisor. Happens often if a field was not initialized (is zero).
5	Addressing Exception: See above
6	Specification Exception: Instruction specific.
7	Data Exception: Generally, a field is being treated as packed decimal when it is not. This is usually a user data error.
8	Fixed Point Overflow Exception: signed binary arithmetic exception controlled by PSW program mask (Floating Point).
9	Fixed Point Divide Exception: See #8
A	Decimal Overflow Exception: See #8
B	Decimal Divide Exception: See #8
C	Exponent Overflow Exception: See #8
D	Exponent Underflow Exception: See #8
E	Significance Exception: See #8
F	Floating Point Divide Exception: See #8
10	Segment Translation Exception. Similar to #5, Addressing Exception, except that the invalid address is within the defined address space, but is marked in the Segment Table as being invalid for read. Hardware does not allow an application to read from storage that has never been written to.
11	Page Translation Exception. Like Segment Translation, except with a page entry.
12	Translation Specification Exception: Error specific to Address Space address resolution.
13	Special Operation Exception: Instruction specific.
40	Monitor Event: Causes a hardware interrupt, and can be intercepted by application exits.

4. In all cases, except #10 and #11, the PSW points past the failing instruction. This is because the hardware instruction counter has already been updated prior to the exception being noticed. With #10 and #11, the PSW points at the failing instruction. msg0S03I contains the actual address of the failing instruction.

Exercise One: Program Check

Analysis:

1. When VSE detects a program check in program mode (bit in PSW), it will take a dump if:

// OPTION NODUMP	No dump will be taken
// OPTION PARTDUMP	Failing partition and selected supv control blocks will be dumped
// OPTION DUMP	Failing partition and entire supervisor will be dumped
// OPTION SYSDUMP	Dump will be taken to VSE System Dump Library (SYSDUMP)
// OPTION DSPDUMP	Data Space will be dumped (see caveats)
// OPTION SYSDUMPC	Dump will not be spooled to SYSLST if SYSDUMP library is full.

2. If the abend occurs in an SVA phase, the failing phase is included in the dump
3. Locate point of foul, and establish module where the error occurred:

- Check for bal registers.
- Check for base registers.
- Check for eyecatcher.

Exercise One: Program Check

Listing:

```
// JOB CSFLDUPD  --EDIT/OVERLAY FIELDS IN C.S.FILE--                 DATE 10/07/1997, CLOCK 20/02/45
// OPTION DUMP
// ASSGN SYS005,897  'I/P KEYFAST TAPE 80/80'
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB
// EXEC CSFLDUPD,SIZE=AUTO
         RUN DATE 10/07/97              C. S. FILE EDIT               -CSFLDUPD-                   PAGE  1
         SOC. SEC. #   JILT   R-CODE   DISH   TELEPHONE#   RATE OF PAY   BIRTH DATE   COPE   -*-   M E S S A G E S
         -----------   ----   ------   ----   ----------   -----------   ----------   ----   -------------------

JOBNAME=CSFLDUPD            DATE=10/07/1997  TIME=20:03:08  CPUID=A0021793 91210180  COMP=56860660615C  PAGE     1

0S03I PROGRAM CHECK INTERRUPTION  - HEX LOCATION 001C3E7A - INTERRUPTION CODE 05 - ADDRESSING EXCEPTION
0S00I JOB CSFLDUPD CANCELED
0S07I PROBLEM PROGRAM  PSW = 071D0000 801C3E7C
0S30I DUMP STARTED. MEMBER=DBG00018.DUMP IN SUBLIB=SYSDUMP.BG
1I49I DUMP LIBRARY FULL

*SYMPTOM RECORD
```

Address of first byte in line

+4 +8 +C +10 +14 +18 +1C

```
00330100                                     00000001 E2D9F9F1 F2F1F0F2 F1F7F9F3  00               ....SR9121021793
00330120 F2F07AF0 F37AF0F8 7AF0F0F9 F761F1F0 61F0F7F0 F07AF0F3 7AF0F87A F0F0F5F6   20:03:08:0097/10/0700:03:08:0056
00330140 F8F6F0F6 F6F0F6F1 F5C30000 E2C3D7D9 C5D84040 00000000 00000000 00400074   860660615C..SCPREQ  ......... ..
00330160 000000B4 002B00B4 000000DF 00000000 00000000 00000000 00000000 00000000   ................................
00330180 00000000 00000000 C1C261E2 F2F0F0F0 40D9C5C7 E261C6C6 C6C6C640 D4E261F0   ........AB/S2000 REGS/FFFFF MS/0
003301A0 E2F0F3C9 40D9C9C4 E261C9D2 D8E5D9D4 40D6C6C6 E261F0F0 F0F0F1C5 F5C340C1   S03I RIDS/IKQVRM OFFS/00001E5C A
003301C0 C261E2F0 F0F0F540 D1D6C26D D5C1D4C5 7EC3E2C6 D3C4E4D7 C440C4E4 D4D7C5C4   B/S0005 JOB_NAME=CSFLDUPD DUMPED
003301E0 6DC4C1E3 C17EC2C7 60D7C1D9 E3C9E3C9 D6D540                                _DATA=BG-PARTITION
```

Exercise One: Program Check

Points past failing instruction

Base register (Reg 7)

```
PSW AND REGISTERS OF ENDING TASK

PSW       071D0000 801C3E7C

GR 0-7    40401238 00000068 00413F38 00000068 00000068 00000004 801C2582 001C3DB8
   8-F    001C3EE0 801C20F2 00410228 00410068 0040AE00 00413B18 801C4392 00000000

AR 0-7    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   8-F    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

FP 0-3    40404040 40404040 40404040 40404040 40404040 40404040 40404040 40404040

CR 0-7    04B1EE40 01000002 00000000 40000000 00000000 01077040 10000000 01000002
   8-F    00000000 00000000 00000000 00000000 00000000 01000002 DF001076 000414E8

IKQVRM    SVA        ADDRESS IS 001C2020  LENGTH IS 000158EB
001C2020 47F0F07E C9D2D8E5 E2D44040 F1F5C340 C4E8F4F3 F6F5F240 F1F161F2 F061F9F6 00 00=IKQVSM  15C DY43652 11/20/96
001C2040 40F5F6F8 F660F0F6 F640C3D6 D7E8D9C9 C7C8E340 C9C2D440 C3D6D9D7 4B40F1F9     5686-066 COPYRIGHT IBM CORP. 19
001C2060 F7F940F1 F9F9F540 C1D3D340 D9C9C7C8 E3E240D9 C5E2C5D9 E5C5C440 D3C9C3C5    79 1995 ALL RIGHTS RESERVED LICE
001C2080 D5E2C5C4 40D4C1E3 C5D9C9C1 D3E260D7 D9D6D7C5 D9E3E840 D6C640C9 C2D490EC    NSED MATERIALS-PROPERTY OF IBM..
001C20A0 D00C18C1 5810FDB0 41110000 89100001 12114780 F09E9680 D00C47F0 F0D089E0    ...A........i.......0.o....00.i.
```

Locate the failing instruction, and back up looking for an eyecatcher.

PSW points here

```
001C3DA0 00000000 00000000 00000000 00000000 00000000 00000000 47F07010 C9D2D8D9    ........................0..IKQR
001C3DC0 E3E54040 F1F5C3F0 1BFF5820 D1745800 C00C9102 C0204780 702A5810 A01447F0    TV  15C0....J.....j...........0
001C3DE0 70364810 D1824811 20044A20 D1809108 C02147E0 70829110 A0284780 70529108    ....Jb......J.j......bj.......j.
001C3E00 B0784710 70524540 70CC4110 00045510 C0144720 70C69102 A0284780 707A9102    .......  .............Fj......:j.
001C3E20 C0204770 707A9108 B0784770 707A4122 00041810 50201000 07FE9110 A02847E0    .....:j......:......&.....j.....
001C3E40 70969108 B0784710 70964540 70CC5840 C0149102 A0284780 70BA9102 C0204770    .oj......o.  ...  ..j.......j.....
001C3E60 70BA9108 B0784770 70BA4122 00044B10 71201514 472070C6 18310E02 07FE41F0    ..j..................F.......0
001C3E80 002307FE 50E0D0C0 18315810 D20C184D 18D141F0 00481A1F 58F0B090 05EF18D4    ....&........K..(.J.0.....0.....M
001C3EA0 58E0D0C0 07FE0000 00000000 00000000 00000000 00000000 00000000 00000000                   ..............
001C3EC0 00000000 00000000 00000000 00000000 00000000 00000000 0004                           .....................
```

Reg 7 is base register

Exercise One: Program Check

Speaker Notes:

1. msg0S03I identifies the failing instruction as being at address x'001C3E7A'. This is the '0E02' (Move Character Long). From register 0 to register 2, with the length of the storage to be moved in registers 1 and 3.
2. The MVCL instruction is interruptible, so the registers in the dump are at the point of failure, not the contents when the instruction was first executed. Register 0 looks like it may have started out with blanks (x'40404040'). This address was immediately invalid, but the hardware updated the registers by the first "chunk" to be moved, before it detected the invalid address.
3. The PSW points just past the 0E02.
4. Once we locate the point of failure, search in the immediate vicinity for a Branch instruction.
5. 472070C6 ➔ Branch Ones to Register 7 + x'C6'.
6. Locate Register 7 in the breakout at the front of the dump. x'001C3DB8' points at the beginning of IKQRTV (A VSE/VSAM module used to retrieve records.)

Exercise One: Program Check

Search in RETAIN for “PROGCK IKQRTV”:

```
** SOFTWARE SUPPORT FACILITY    TITLE PAGE      1     **

LIB/FILE(S) CURRENTLY SELECTED DS/AC
PROGCK  IKQRTV

THE ABOVE SEARCH ARGUMENT RESULTED IN     3 MATCHES
USER PTF#,APAR#,ABS.L1                            (APAR DEFAULTS)

    1 UD47798 DY41632 PROGCK IN VSAM USING IIP (ISAM INTERFACE)
    2         DY41110 PROGCK IN IKQRTV DURING GET PROCESSING WIT
    3         II08262 USING THE ISAM INTERFACE PROGRAM (IIP) RES
```

APAR: II08262

-
-
-

ERROR DESCRIPTION:
Using the ISAM Interface Program (IIP) from old (backlevel) COBOL programs, may result in progck in VSAM due to invalid 31bit addresses passed to VSAM in the RPL. COBOL is using the high order byte of those fields in the DTFIS (IOASAD2, KARGAD2) that are used by IIP to modify the according RPL fields (RPLAREA and RPLARG).

Exercise One: Program Check

Since most of the customers have no source for these programs and the COBOL release can no longer be maintained, VSAM provides the following patch to phase IIPOPEN to clear the high order byte, when the addresses are moved from the second to the first part of the DTFIS:

```
// JOB PATCH                                          APAR= II08262
// EXEC MSHP,SIZE=900K
 PATCH S=IJSYSRS.SYSLIB
 AFFECTS PHASE=IIPOPEN
 ALT 00EE D20380747008 : D20280757009   /* move ioarea   addr */
 ALT 00F4 D20380707014 : D20280717015   /* move workarea addr */
 ALT 00FA D20380607010 : D20280617011   /* move key arg  addr */
 ALT 0218 D20380747008 : D20280757009   /* move ioarea   addr */
 ALT 021E D20380707014 : D20280717015   /* move workarea addr */
 ALT 0224 D20380607010 : D20280617011   /* move key arg  addr */
/*
/&
```

The symptoms of the ABEND in VSAM code are most commonly a PROGCK in phase IKQVRM, modules IKQIXS or IKQRTV or similar.

Determine VSE System Status

Determine System Status:

- System disabled wait state (hard wait)
- System disabled loop
- System enabled wait state (soft wait)
- Partition wait
- Partition loop

Determine VSE System Status

1. Is Attention Routine responding?

   - Immediate command:

```
REPLID
AR 0015 1I88I NO REPLIES OUTSTANDING
```

   Response? No… go to page **17** to check for hard wait or disabled loop.

2. Reply to all outstanding messages, then attempt a more complex command:

```
VOLUME 140
AR 0015 CUU  CODE DEV.-TYP   VOLID  USAGE   SHARED    STATUS    CAPACITY
AR 0015 140  6E    3390-3   DOSRES  USED                             1112  CYL
AR 0015 1I40I  READY
```

   Response? No…Enter "**RC**" and retry command. Still no response? Look for console management or vendor product problems

3. Are higher priority partitions running?

```
PRTY
AR 0015 PRTY Z,Y,P,C,BG,FB,FA,F9,F8,F7,F6,F5,F4,F2,F3,F1
```

POWER:

```
     D T
     AR 0015 1C39I COMMAND PASSED TO VSE/POWER
     F1 0001 1R46I  TIME IS 17:23:07, DATE IS 07/31/2002
     F1 0001 1R46I  019 PAGES FIXED, 023 CURRENT TASKS
```

CICS:

Determine VSE System Status

```
MSG F2
F2-0100
100 CEMT I TA
F2-0100
F2 0103
    Tas(0000023) Tra(CXPB)           Sus Tas Pri( 001 )
    Sta(S ) Use(DBDCCICS) Rec(X'B7F46C3740692D02') Hty(OPEN_ANY)
    Tas(0000025) Tra(ICVS)           Sus Tas Pri( 001 )
.
.
.
    RESPONSE: NORMAL TIME:  17.25.52  DATE: 07.31.02
    SYSID=CIC1 APPLID=DBDCCICS
100
```

VTAM:

```
D NET,MAJNODES
AR 0015 1C39I COMMAND PASSED TO ACF/VTAM
F3 0003 IST097I DISPLAY ACCEPTED
F3 0003 IST350I DISPLAY TYPE = MAJOR NODES
F3 0003 IST089I VTAMSEG  TYPE = APPL SEGMENT     , ACTIV
F3 0003 IST089I VTAMSEG  TYPE = APPL SEGMENT     , ACTIV
F3 0003 IST089I NODE0001 TYPE = PU T4/5 MAJ NODE , ACTIV
F3 0003 IST089I ISTPDILU TYPE = CDRSC SEGMENT    , ACTIV
F3 0003 IST089I ISTADJCP TYPE = ADJCP MAJOR NODE , ACTIV
```

Respond? No … Transfer analysis to higher priority partition.

Determine VSE System Status

4. Check status of failing partition:

STATUS F2

```
AR 0015 M23 F2 CICSICCF 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     TCB=00056A60 TIB=0004D600 SAV=00500000
AR 0015     SCB=0004A70C PCB=0004B100 COM=00003BD8
                              -or-
                         82 WAITING FOR I/O ON DEVICE=009
                              -or-
                         82 WAITING ON TIMER INTERRUPT
                              -or-
                         82 WAITING FOR OPERATORS RESPONSE
                              -or-
                         83 READY TO RUN
                              -or-
                         85 WAITING FOR PROGRAM FETCH
                              -or-
                         8C WAITING FOR SUB-TASK TERMINATION
                              -or-
                         8E WAITING FOR LOCKED RESOURCE USED BY <task>
                              -or-
                         95 WAITING FOR DUMP/TRACE PROCESS   USED BY F4
                              -or-
                         57 VSE/POWER WAITING FOR WORK
```

Determine VSE System Status

- If "**WAITING FOR I/O, ECB OR TECB**", partition is waiting for an event to occur.
  See "Exercise Two, Partition in I/O Wait" on page 17.
- If "**READY TO RUN**", partition may be in a loop.
- If "**WAITING FOR I/O ON DEVICE=<cuu>**"
  i. Check using the IUI System Status screen if partition is performing I/O.
  ii. Check status of device using "**STATUS <cuu>**".
  iii. Save partition, SVA, and supervisor to tape using "**DUMP**" command
  iv. Try "**CANCEL <cuu>**". If this doesn't work, try "**CANCEL <cuu>,FORCE**".
  Caution: A partition may be doing productive work (i.e. copying large amounts of data) and still show waiting on I/O each time it is checked.
- If "**WAITING FOR LOCKED RESOURCE USED BY <task>**", check status of "**<task>**", which owns the resource.
- If any indication that the partition may be waiting on POWER spooled device, also enter "**D A**" and "**D TASKS**".
- If "**VSE/POWER WAITING FOR WORK**", previously running application has canceled, and partition is paused at end of job boundary. Check for previous error messages in this partition.

Determine VSE System Status

5. Is partition in a wait or loop? Check System Utilization using IUI:

```
IESADMSL.IESEADM      VSE/ESA FUNCTION SELECTION      APPLID: DBDCICS
Enter the number of your selection and press the ENTER key:
         1  Installation
         2  Resource Definition
   ==>   3  Operations
         4  Problem Handling
         5  Program Development
         6  Command Mode
         7  CICS-Supplied Transactions

 PF1=HELP                    3=SIGN OFF                  6=ESCAPE(U)
                             9=Escape(m)
==>
```

Select "**OPERATIONS**", "**SYSTEM STATUS**":

```
IESADMSL.IESEOPS            OPERATIONS                 APPLID: DBDCCICS
Enter the number of your selection and press the ENTER key:
        1  Console
        2  Manage Batch Queues
        3  Display Active Users/Send Message
        4  Enter News
        5  Retrieve Message
   ==>  6  System Status
        7  Backup/Restore
        8  Personal Computer Move Utilities
        9  Transfer Files and Jobs to Another System
```

Select "6 **SYSTEM STATUS**":

```
IESADMSL.IESESTAT           SYSTEM STATUS              APPLID: DBDCCICS
```

Determine VSE System Status

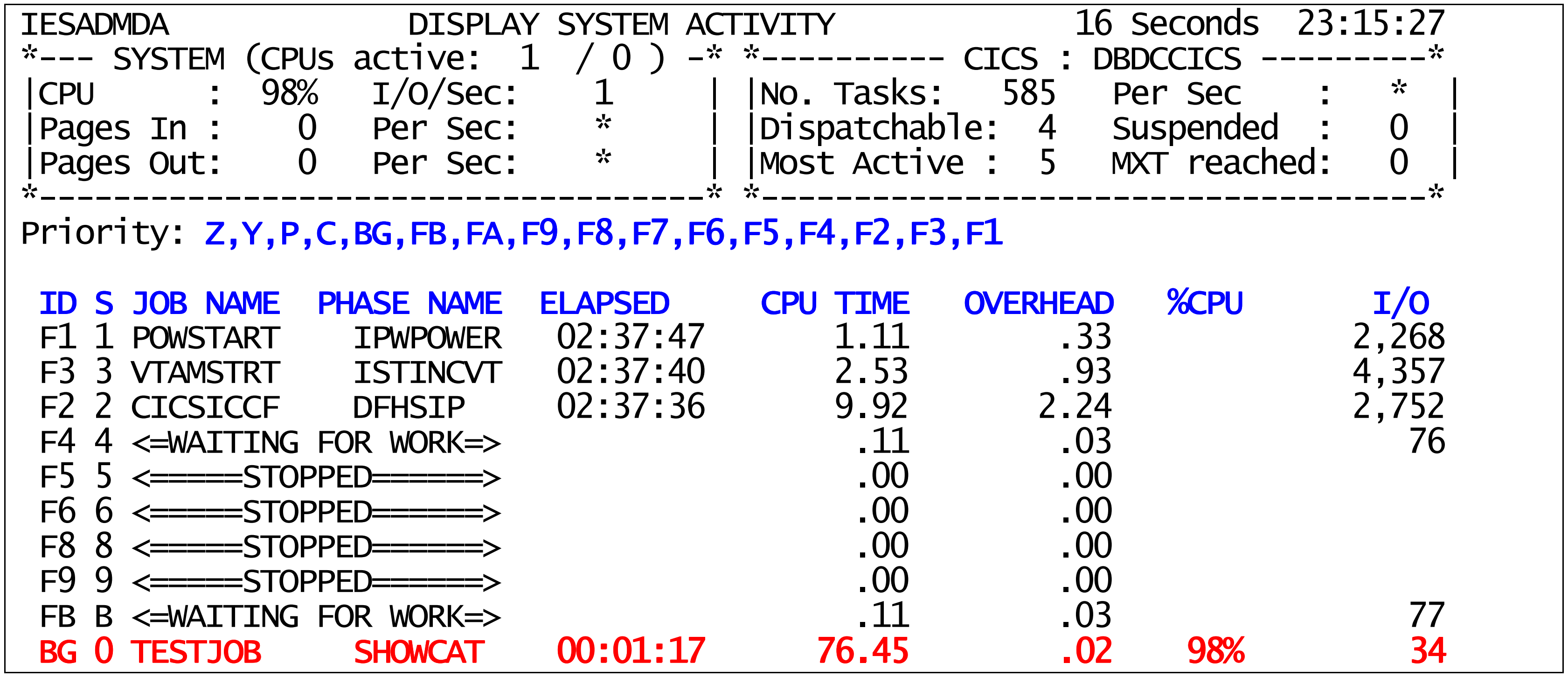
```
Enter the number of your selection and press the ENTER key:
   ==>  1  Display System Activity
        2  Display Channel and Device Activity
        3  Display Storage Layout
        4  Display CICS TS Storage
```

Select "**DISPLAY SYSTEM ACTIVITY**":

```
IESADMDA             DISPLAY SYSTEM ACTIVITY            15 Seconds  23:14:11
*--- SYSTEM (CPUs active:  1  / 0 ) -* *---------- CICS : DBDCCICS ---------*
|CPU       :    *   I/O/Sec:    3     | |No. Tasks:   580   Per Sec    :   *  |
|Pages In  :    0   Per Sec:    *     | |Dispatchable:  4   Suspended  :   0  |
|Pages Out:     0   Per Sec:    *     | |Peak Active :  5   MXT reached:   0  |
*-------------------------------------* *-------------------------------------*
Priority: Z,Y,P,C,BG,FB,FA,F9,F8,F7,F6,F5,F4,F2,F3,F1

 ID S JOB NAME   PHASE NAME  ELAPSED     CPU TIME   OVERHEAD   %CPU       I/O
 F1 1 POWSTART     IPWPOWER   02:36:31        1.11        .33             2,268
 F3 3 VTAMSTRT     ISTINCVT   02:36:24        2.52        .93             4,352
 F2 2 CICSICCF     DFHSIP     02:36:20        9.86       2.24             2,752
 F4 4 <=WAITING FOR WORK=>                     .11        .03                76
 F8 8 <=====STOPPED======>                     .00        .00
 F9 9 <=====STOPPED======>                     .00        .00
 FA A <=====STOPPED======>                     .00        .00
 FB B <=WAITING FOR WORK=>                     .11        .03                77
 BG 0 TESTJOB      SHOWCAT    00:00:01        1.59        .00                34
```

Determine VSE System Status



```
IESADMDA                  DISPLAY SYSTEM ACTIVITY             16 Seconds  23:15:27
*--- SYSTEM (CPUs active:  1  / 0 ) -* *---------- CICS : DBDCCICS ---------*
|CPU        :  98%   I/O/Sec:    1    | |No. Tasks:   585   Per Sec    :   *  |
|Pages In :     0   Per Sec:    *    | |Dispatchable:  4   Suspended  :   0  |
|Pages Out:     0   Per Sec:    *    | |Most Active :  5   MXT reached:   0  |
*-------------------------------------* *-----------------------------------*
Priority: Z,Y,P,C,BG,FB,FA,F9,F8,F7,F6,F5,F4,F2,F3,F1

 ID S JOB NAME   PHASE NAME  ELAPSED     CPU TIME    OVERHEAD   %CPU        I/O
 F1 1 POWSTART    IPWPOWER   02:37:47        1.11         .33             2,268
 F3 3 VTAMSTRT    ISTINCVT   02:37:40        2.53         .93             4,357
 F2 2 CICSICCF    DFHSIP     02:37:36        9.92        2.24             2,752
 F4 4 <=WAITING FOR WORK=>                    .11         .03                76
 F5 5 <=====STOPPED======>                    .00         .00
 F6 6 <=====STOPPED======>                    .00         .00
 F8 8 <=====STOPPED======>                    .00         .00
 F9 9 <=====STOPPED======>                    .00         .00
 FB B <=WAITING FOR WORK=>                    .11         .03                77
 BG 0 TESTJOB     SHOWCAT    00:01:17       76.45         .02   98%          34
```

If partition is in a loop (logging CPU time), record a few PSWs from the current TCBSAVE area prior to canceling partition: **CANCEL BG,DUMP**.
If this does not cancel the job, try: **CANCEL BG,DUMP,FORCE**

Determine VSE System Status

6. We have already established that the system is not responding.
   a. Is system in hard (disabled) wait or disabled loop?
   b. Locate the current system PSW.
      Either from the integration console, or, if under VM/ESA, via a "CP D P" command.

PSW: (Program Status Word, controls instruction execution)

```
07BD2000 82ABB0B2
```

'D' indicates that the wait bit is OFF.

'07' indicates that Interrupts are enabled.

```
000A0000 00001000
```

'A' ➔ Wait bit is on.

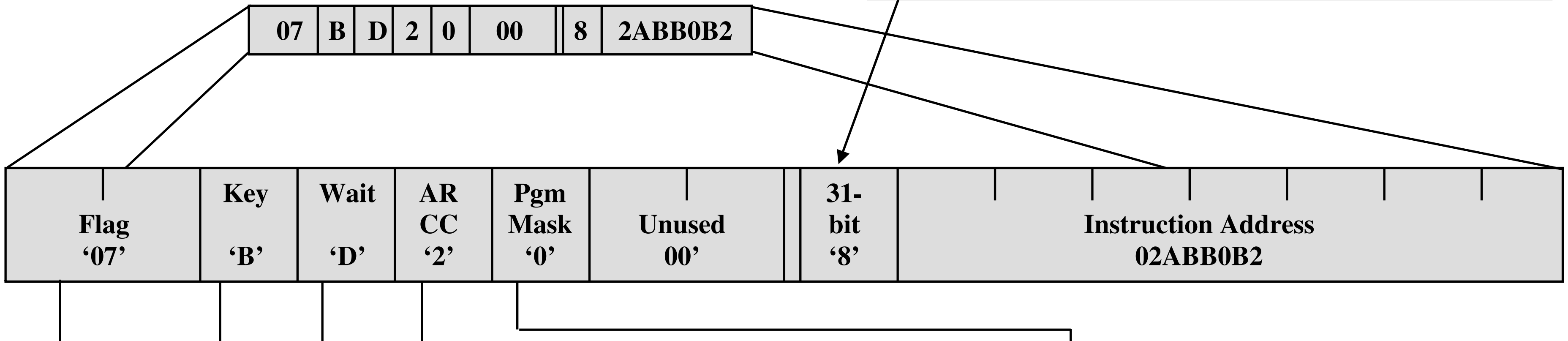
'00' ➔ Interrupts are disabled.

Determine VSE System Status

Speaker Notes:

07BD2000 82ABB0B2

‘8’ If instruction address begins with x’8’, machine is operating in AMODE(31) (31-bit addressing mode)



Programming Mask:

‘8’	Fixed-point overflow mask
‘4’	Decimal overflow mask
‘2’	Exponent underflow mask
‘1’	Significance mask

‘8’	Extended Control (EC) Mode
‘4’	Machine Check interrupts enabled
‘2’	Wait state
‘1’	Problem State

‘0’ = Supervisor, ‘1’ = BG, ‘2’ = ‘FB’, ‘3’ = FA’, ‘4’ = F9 … **‘B’ = F2**, ‘C’ = F1, ‘D’ = Dynamic Partition.

‘80’	Unused
‘40’	Program Event Recording (PER) enabled
‘20’ – ‘08’	Unused
‘04’	Translation (virtual) mode
‘02’	Input/Output interrupts enabled
‘01’	External interrupts enabled

Address Space Control (AS)

’00’ =	Primary mode
‘10’ =	Access Register (AR) mode, Home Space
’11’ =	Access Register mode, Secondary Space

Condition Code (CC)

’00’ =	Zero Both operands equal	’10’ =	Greater than zero First operand high Carry
’01’ =	Not zero Less than zero First operand low	’11’ =	Not Zero Overflow

Determine VSE System Status

Hard Wait:

From the hardware console, place the system in stop mode.

If unable to stop the machine, you may be in a disabled PSW load loop
(e.g. if low core is overlaid and Progck New PSW is invalid).

Check the current PSW. Is wait bit on? No, it is probably a disabled loop (see next page).

Hardwait = PSW wait bit (14) is on, and interrupt masks (bits 6&7) are off

x’03’ = zero = interrupts disabled
x’02’ = on = wait bit

000A0000 00001000	Disaster error detected by VSE/AF supervisor
000A0000 00001122	Abend while processing program check (Normally indicates overlay of supervisor)
000A0000 00CExxxx	Stand-alone dump program. xxxx is rc.
010E0000 0000EEEE	SDAID address stop or DEBUG stop (can be restarted by pressing external interrupt)
010E0000 00EEEEEE	SDAID intervention required (can be restarted by making device ready)
040E0000 00002000	Stand-alone supervisor, normal completion

Determine VSE System Status

Speaker Notes:

1. Check low core. Screen print it. If there is no hardwait code in low core, but the system still does not respond to attention routine commands, we may be in a disabled loop.

   See *VSE/ESA Messages and Codes, Volume 2* under "VSE/Advanced Functions Codes and SVC Errors" for a complete list of hardwait codes.

x'0FFF'	Progck in supervisor mode
x'0FFE'	I/O Error during fetch from IJSYSRS.SYSLIB
x'0FED'	Inconsistency (logic error) detected in supervisor
x'0FFB'	Page Fault during non-pageable supervisor activity
x'0FF9'	Error during Page I/O

   Special codes are placed in low core if errors are detected during IPL, including:

x'C1 E2'	Machine check on clear storage
x'07 E6 cc uu'	IPL input/output error (ccuu is device where error occurred). I/O error sense information will also be stored in low core.
x'07 E6 C3 E2'	Console router error ('CS')
x'07 E6 C9 C3'	Integrated Console error ('IC')
x'cc 00 0F D0'	Error during IPL (cc is supervisor cancel code)
x'F0 C9 F0 F0 C1' '0I00A'	… Low Core may contain message documented in "*VSE/ESA Messages and Codes, Volume 1*"

Determine VSE System Status

Documentation for Hard Wait

1. Check low core. Save the first x’100’ bytes.
2. Perform Store Status (saves PSW and registers in low core)
3. Mount stand-alone tape and ipl the tape.
4. When you receive message “Standalone dump complete”, re-ipl VSE.
5. Load stand-alone tape using IUI.
6. Perform analysis of stand-alone dump.
7. Locate failing application, if not identified in previous step.
8. Contact owner of failing application (IBM, Vendor, local support group)

Determine VSE System Status

Disabled Loop:

1. Screen print the first x’100’ bytes of low core.
2. Write down current PSW.
3. Continue with step 3 under “Hard Wait”.
4. If low core is overlaid, this can cause a disabled PSW load loop. Depending on the processor, it may not be possible to stop the CPU to take a standalone dump. In this case, write down (or print) as much of low core as possible.

Exercise Two: Partition in I/O Wait

Exercise 2:

1. Identify the last instruction executed by a partition in a wait state

2. Identify the module where the last instruction occurred using:
    - Base Registers
    - Branch and Link registers
    - Scanning for an eyecatcher.

Documentation:

- If a loop, record a few instructions from TCBSAVE.
- Console log containing analysis.
- AR DUMP of failing partition
- AR DUMP of supervisor and SVA
- If waiting on locked resource, AR DUMP of partition which owns the resource.
- CANCEL <partition>,DUMP

Exercise Two: Partition in I/O Wait

Speaker Notes:

If partition is in a wait state, PSW usually points to a Supervisor Call (SVC = x’0Ann’):
Otherwise, partition may be in a loop, or waiting on page I/O.

`‘0A00’`	=	Start I/O (Register 1 points at CCB)
`‘0A01′`	=	Fetch a phase (Register 1 points at phase name)
`‘0A02′`	=	Fetch a $B transient (Register 1 points at phase name. If “$$BOPEN” or “$$BCLOSE”, Register 0 points a list of files to be processed)
`‘0A04′`	=	Load a phase and return entry address (See SVC 01)
`‘0A06′`	=	Cancel
`‘0A07′`	=	Wait for I/O (See SVC 00) Note: PSW is backed up 10 bytes on SVC 07
`‘0A0E′`	=	End of Job
`‘0A29′`	=	Dequeue (Register 1 points at Resource Control Block = RCB. RCB+x’04 points at ECB to be freed)
`‘0A2A′`	=	Enqueue (See SVC x’29’)
`‘0A3D′`	=	Request getvis storage (Register 0 contains length of area to be acquired)
`‘0A3E′`	=	Free storage (Register 1 contains pointer to area to be freed. Register 0 contains length of area to be freed.)
`‘0A41′`	=	CDLOAD (load phase into GETVIS and return address) (See SVC 01)
`‘0A6E’`	=	Lock / Unlock a resource (Register15 = x’……03’ → Lock request, Register 15 = x’……01’ → Unlock request Register 1 points at DTL = Define The Lock DTL+ 4 (length = 12) contains name of lock)
`‘0A71′`	=	Cross Partition Communication Control (XPCC) (Register 1 points to CRCB +x’08’ = Path id, connection token)

‘0A01’, ‘0A02’, ‘0A04’, or ‘0A41’ would indicate waiting on program fetch.

Exercise Two: Partition in I/O Wait

Analysis:

1. Using status command, determine status of partition, and locate main task Task Control Block (TCB).
2. Locate current task save area in TCBSAVE
3. Locate failing PSW.
4. Use base register, or backup from point of impact, looking for a module eyecatcher.

Normally, the PSW points just past the last instruction to be executed. There are two exceptions:

1. Program check for segment translation exception (x'10') or page translation exception (x'11'). In this case, the PSW points at the failing instruction.
2. VSE Standard Wait coding, after a Wait on ECB or I/O (SVC 07):

   ```
   91801002    TM  CCBFLAG,POSTED
   4710rddd    BO  *+6
   0A07        SVC 07
   ```

   If I/O is not yet complete, VSE backs up PSW by 10 bytes, back to Test-Under-Mask

Exercise Two: Partition in I/O Wait

Problem Description:

VSE/ESA 2.2: Customer installed maintenance for Power in PMR 48228 B550 and now he has jobs that look to stall out. They do not get any I/O or CPU time. They can purge the jobs then restart them and they will run. His Cobol programs calling OEM SoftwareAG look to be the jobs that have problem. Customer request call back today.

Analysis:

```
STATUS
AR 0015    T1F S1F       WAITING FOR I/O, ECB OR TECB
AR 0015    T20 AR        READY TO RUN
AR 0015    T21 BG        VSE/POWER WAITING FOR WORK
AR 0015    T4D     -F1   WAITING FOR I/O, ECB OR TECB
AR 0015    T4E     -F1   WAITING FOR I/O, ECB OR TECB
AR 0015    T55     -F1   WAITING FOR I/O, ECB OR TECB
AR 0015    T22 F1        WAITING FOR I/O, ECB OR TECB POWER MAIN TASK
AR 0015    T72     -F2   WAITING FOR I/O, ECB OR TECB
AR 0015    T73     -F2   WAITING FOR I/O, ECB OR TECB
AR 0015    T23 F2        WAITING FOR I/O, ECB OR TECB
AR 0015    T4F     -F3   WAITING FOR I/O, ECB OR TECB
AR 0015    T50     -F3   WAITING FOR I/O, ECB OR TECB
AR 0015    T51     -F3   WAITING FOR I/O, ECB OR TECB
AR 0015    T52     -F3   WAITING FOR I/O, ECB OR TECB
      • • •
AR 0015 1I40I  READY
```

Exercise Two: Partition in I/O Wait

D TASKS

```
AR 0015 1C39I COMMAND PASSED TO VSE/POWER
F1 0001 1R48I  ***   BEGIN OF DISPLAYING VSE/POWER TCB'S       ***
F1 0001 1R48I  TID ,CUU,TCBADR,T,PHASE(ADDR),REG12 ,STATE(DETAIL)
F1 0001 1R48I  LSNA,    ,509A40, ,SN (5A0D00),5A0D5E,M(R01=3D509C38)
F1 0001 1R48I  LLDR,    ,50B000, ,LD (5AD000),5AD304,C(R01=3D50B034)
F1 0001 1R48I  YTES,    ,502B40, ,TV (59C700),59C7E0,C(R01=3D502B74)
F1 0001 1R48I  XMAS,    ,505A20, ,XM (58FC00),58FF26,M(R01=3D505BB8)
F1 0001 1R48I  JSPM,    ,506820, ,SM (59CB00),59CD08,Q(R01=3D2EF0B8)
F1 0001 1R48I  O CP,    ,5012A0, ,CD (531400),534376,R(----------)   (D TASKS)
F1 0001 1R48I  NTFY,    ,50C560, ,NS (598700),598C24,M(R01=3D50C5F4)
F1 0001 1R48I  NRV ,SNA,50AB40, ,BS (5BF200),5BF2E6,B(NCB=3D509CC0) (PWR01,CON)
F1 0001 1R48I  DPST,    ,503D80, ,DP (575600),5756EE,M(R01=3D503BE0)
F1 0001 1R48I  E BG,FEC,507000, ,XRE(54EA00),54FC36,Q(R01=3D507020)
F1 0001 1R48I  E F2,FEC,507280, ,XRE(54EA00),54F844,C(R01=3D5072B4)
     • • •
F1 0001 1R48I  ***      END OF DISPLAYING VSE/POWER TCB'S      ***
```

D A

```
AR 0015 1C39I COMMAND PASSED TO VSE/POWER
F1 0001 1R48I   C-RV ,SNA,   AWAITING     NODE=3DPWR01
F1 0001 1R48I   F2,FEC,2A,   ADA5SRC ,30667,A
F1 0001 1R48I   F3,FEC,3V,   VTAMSTRT,22872,3
F1 0001 1R48I   F4,FEC,4A,   ADA5AUT ,30670,A
F1 0001 1R48I   F5,FEC,5A,   ADA5MDL ,30673,A
F1 0001 1R48I   F6,FEC,6A,   INACTIVE,
     • • •
```

Exercise Two: Partition in I/O Wait

STATUS P1
```
AR 0015    T34 P1 DRP400 82 WAITING FOR I/O, OR ECB POSTING
AR 0015        TCB=003722E8 TIB=00372268 SAV=00540000
AR 0015        SCB=00372000 PCB=00372088 COM=003724F0

AR 0015 1I40I  READY
```

Save area:(“SAV=” points at partition save area)

SHOW P1,540000.70
```
540000         0    C4D9D7F4 F0F04040 07DD0000 0011C9BC | DRP400  ......I. |
540010        10    005DB700[9] 00583F80[A] 0011CB07[B] 0011BB08[C] | .)............... |
540020        20    00584748[D] 9011C3C2[E] 0011BB08[F] 00585432[0] | ......CB........ |
540030        30    005845F8[1] 00584630[2] 00000085[3] 005845F8[4] | ...8.......e...8 |
540040        40    50585588[5] 00000420[6] 00584080[7] 00584630[8] | ...h...... ..... |
540050        50    0000AF75 6AE55569 40404040 40404040 | .....V..         |
540060        60    40404040 40404040 40404040 40404040 |                  |
```

PSW: Shows last instruction executed

Exercise Two: Partition in I/O Wait

PSW, however, is not pointing in the partition, it is pointing into the SVA:

```
11BB00        0    001AF1F9 F2F0F5F0 47F0F0BA 47F0F0BE | ..192050.00..00. |
11BB10       10    47F0F0B6 47F0F0C2 47F0F0B6 47F0F0B6 | .00..00B.00..00. |
11BB20       20    47F0F0B6 47F0F098 47F0F0A6 5BC9D1C4 | .00..00q.00w$IJD |
11BB30       30    D7D9E340 C3F4F4F0 C4C1F4F1 F6F8F440 | PRT C440DA41684  |
11BB40       40    F5F6F8F6 60F0F0F7 404DC35D 40C3D6D7 | 5686-007 (C) COP |
11BB50       50    E8D9C9C7 C8E340C9 C2D440C3 D6D9D740 | YRIGHT IBM CORP  |
11BB60       60    F1F9F8F3 6BF1F9F8 F940C1D3 D340D9C9 | 1983,1989 ALL RI |
11BB70       70    C7C8E3E2 40D9C5E2 C5D9E5C5 C440D3C9 | GHTS RESERVED LI |
11BB80       80    C3C5D5E2 C5C440D4 C1E3C5D9 C9C1D3E2 | CENSED MATERIALS |
    •
    •
    •
11C990       90    07FE50E0 91E45860 90D8D501 605A908C | ....jU.-.QN.-!.. |
11C9A0       A0    4780CEB0 41109060 0A009180 10024710 | .......-..j..... |
11C9B0       B0    CEAC0A07 47F0CEBE 581090E0 91801002 | .....0......j... |
11C9C0       C0    4710CEBE 0A079508 908B4770 CEF29180 | ......n......2j. |
11C9D0       D0    909047E0 CEF29101 909247E0 CEDE5860 | .....2j..k.....- |
```

PSW points here

Reg 12 points here

Branch shows Reg12 to be base reg

CCB (Reg1): It is not posted, but is complete (x'0C').

```
5845F8        0    00000001 0C000114 005DB710 005DB718 | .........)...).. |
584608       10    0011BB08 08B40909 00584631 00000000 | ................ |
584618       20    07004120 E0000000 095846C1 20000084 | ...........A...d |
584628       30    00000000 00000000 014040F5 4B40D9C5 | .........  5. RE |
```

Exercise Two: Partition in I/O Wait

Search in RETAIN for “$ijdprt wait”:

```
** SOFTWARE SUPPORT FACILITY    TITLE PAGE       1     **
LIB/FILE(S) CURRENTLY SELECTED DS/AC
$IJDPRT  WAIT

THE ABOVE SEARCH ARGUMENT RESULTED IN     2 MATCHES
USER PTF#,APAR#,ABS.L1                                         (APAR DEFAULTS)

    1         DY41428 VSERAS
    2 UD50251 DY44442 SERVICE RELEASE 04/1997 FOR VSE/ESA VERSIO

PROBLEM SUMMARY:                                                   ACTIVE
****************************************************************  OPTIONS ARE:
* USERS AFFECTED: All.                                         *
****************************************************************
* PROBLEM DESCRIPTION: 1. SVC07 Softwait with parallel POWER   *
*                      2. ILLEGAL SVC with SVC03 fct code 06   *
*                      3. IJBAR Enhancements:                  *
*                         A. GETVIS SVA,DETAIL                 *
*                         B. SIR SMF,cuu                       *
****************************************************************   HIT       2
* RECOMMENDATION:                                              *   OF        2
**************************************************************** APAR= DY44442
1. Softwait with an SVC07 (WAIT) in module $ijdprt with Turbo    OWNED BY: RS3
   Dispatcher active and VSE/POWER running in parallel mode.
   The dump shows a COBOL program that is waiting for a spooled
   print I/O. The traffic bit in the CCB is not posted, although
   the status looks like I/O complete.
```

Exercise Three: Partition in Lock Wait

Exercise 3:

A good deal of this analysis can be done directly from the console:

1. Identify status of hanging partition.
2. Identify the last instruction executed by the partition
3. Identify the module where the last instruction occurred using:
   - Base Registers
   - Branch and Link registers
   - Scanning for an eyecatcher.
4. For "Waiting on a Lock" wait state, Identify the name of the lock.

Documentation:

- Console log containing analysis.
- AR DUMP of failing partition
- AR DUMP of supervisor and SVA
- If waiting on locked resource, AR DUMP of partition which owns the resource.
- CANCEL <partition>,DUMP

Exercise Three: Partition in Lock Wait

Problem Description:

VSE/ESA 2.6: Partition hangs during EOT (End of Task). Until partition is flushed, other partitions hang during VSE/VSAM Open.

Analysis:

Owner of lock

1. Identify status of hanging partition.

```
STATUS F8
AR 0015 M29 F8 LOADKSDS 8E WAITING FOR LOCKED RESOURCE        USED BY F7
AR 0015    TCB=00057770 TIB=0004FB80 SAV=00500000
AR 0015    SCB=0004D2A4 PCB=0004E4C0 COM=00004358
AR 0015 1I40I  READY
```

2. Identify the last instruction executed by the partition

Partition Save Area

```
SHOW F8,500000.50
AR 0015    DATA  FOUND AT 00500000
V00500000 D3D6C1C4 D2E2C4E2 075D1000 00165E44 56 *LOADKSDS.)....;.* R0348B000
V00500010 008831B9[9] 008838CC[A] 00882100[B] 401650DA[C] 56 *.h...h...h.. .&.* R0348B010
V00500020 00882528[D] 40165ADC[E] 00000000[F] 0000000B[0] 56 *.h.. .!.........* R0348B020
V00500030 00881B80[1] 008838CC[2] 00883018[3] 001660DA[4] 56 *.h...h...h....-.* R0348B030
V00500040 00881B80[5] 00883027[6] 00000005[7] 00000001[8] 56 *.h...h..........* R0348B040
AR 0015 1I40I  READY
```

Exercise Three: Partition in Lock Wait

3. Identify the module where the last instruction occurred:

```
V001650A0 ........ ........ 47F0F01A 14C9C7C7 06 *.........00..IGG* R0062A0A0
V001650B0 F0C3D3C1 C840C4C1 F540C4E8 F4F1F7F0 06 *0CLAH DA5 DY4170* R0062A0B0
V001650C0 F0000000 C9C7C7D7 E2C3C140 47F0FFA6 06 *0...IGGPSCA .0.w* R0062A0C0
V001650D0 90CED00C 41DD000C 05C0414C 0FFF4144 06 *..}......{.<....* R0062A0D0
V001650E0 00015850 B0149180 500247E0 C05C9160 06 *...&..j.&..\{*j-* R0062A0E0
V001650F0 50024770 C05C95E4 50124770 C05C9118 06 *&...{*nU&...{*j.* R0062A0F0
     .
     .
     .
V00165E30 58A0A028 D2055010 A07494DE 50034100 06 *....K.&...m.&...* R0062AE30
V00165E40 000B0A6E 12FF4780 CD8A49F0 40124780 06 *...>.......0 ...* R0062AE40
V00165E50 CD8A40F0 B22E92F6 B004D201 B2884086 06 *.. 0..k6..K..h f* R0062AE50
```

PSW points here

Reg12 = Base Reg

Reg1 => DTL

```
SHOW F8,00881B80
AR 0015   DATA  FOUND AT 00881B80
V00881B80 001E1108 E5E2E8E2 D6D7C5D5 00000000 56 *....VSYSOPEN....* R03492B80
V00881B90 E2E8E2E6 D2F30000 00000000 00000000 56 *SYSWK3..........* R03492B90
```

Exercise Three: Partition in Lock Wait

Task Info Block (TIB)
(See STATUS output)

SHOW 0004FB80.50

```
AR 0015    DATA   FOUND AT 0004FB80
V0004FB80 8005ADE4 0005D6D5 00057770 000000                    * R0004FB80
V0004FB90 00000000 00290050 0004E4C0 80000000 06 *.......&..U{....* R0004FB90
V0004FBA0 0004D2A4 00000000 40000000 8E000000 06 *..Ku.... .......* R0004FBA0
V0004FBB0 00000000 00000000 0004D2A4 00000000 06 *..........Ku....* R0004FBB0
V0004FBC0 00000000 00000000 00000000 00000000 06 *................* R0004FBC0
AR 0015 1I40I   READY
```

Lock Element

SHOW 0005D6D4.50

```
AR 0015    DATA   FOUND AT 0005D6D4
V0005D6D0 ........ 030CEB30 00000000 E5E2E8E2 06 *............VSYS* R0005D6D0
V0005D6E0 D6D7C5D5 00000000 11880001 0005D6F4 06 *OPEN.....h....O4* R0005D6E0
V0005D6F0 00000000 030CEEB0 00000000 C4E3E2E5 06 *............DTSV* R0005D6F0
AR 0015 1I40I   READY
```

Owner Element

SHOW 030CEB30

```
AR 0015    DATA   FOUND AT 030CEB30
V030CEB30 00000000 00280000 00011000 00000000 06 *................* R00791B30
V030CEB40 00000000 00230001 00000000 00000000 06 *................* R00791B40
```

x‘28’ = F7

Exercise Four: I/O Error

Exercise 4:

1. Interpret hardware error indicators
   in VSE/ESA console messages.
2. Locate CCB in I/O Error partition dump.
3. Locate failing CCW.
4. Identify dasd address

Is anyone asleep yet?

Exercise Four: I/O Error

CCB: (Command Control Block)

Created by the application (VSE/VSAM for instance) to request Input/Output by VSE)

```
00009800 0C000124 00631280 00631298
```

First CCW in chain

x’0C’ = I/O complete (copied from CSW)

x’80’ = CCB posted complete (by VSE)

CCW: (Channel Command Word)

The CCB points to CCWs, each of which specifies a single I/O command

```
06400010 00088638
```

Address where hardware is to place the data

x’06’ = Read data

CSW: (Channel Status Word)

Passed to hardware by VSE via SSCH command.
Returned by hardware when I/O complete.

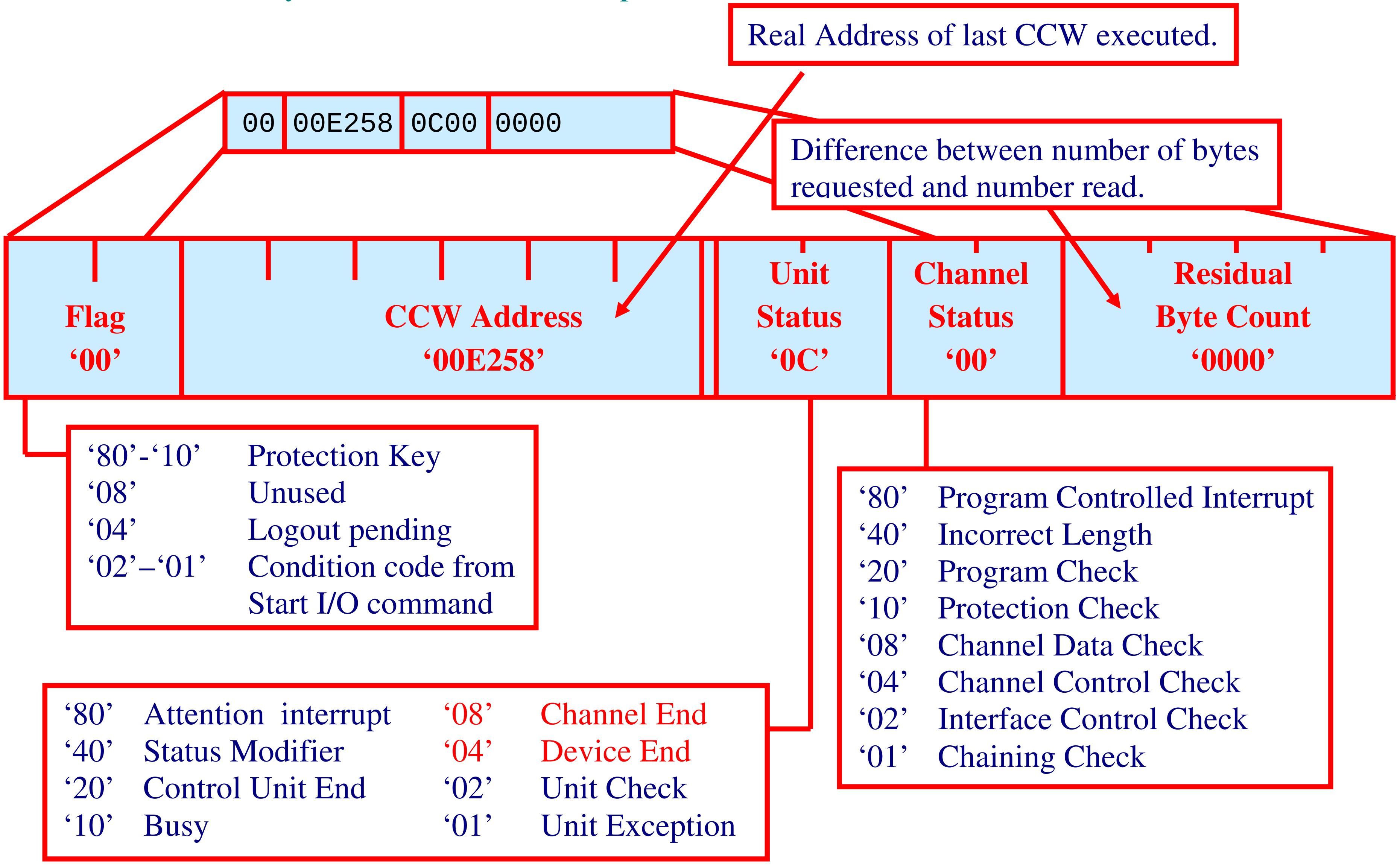
```
0000E258 0C000000
```

x’0C’ = I/O complete (posted by hardware)

Address of last CCW executed

Exercise Four: I/O Error

CSW:(Channel Status Word) passed to hardware by VSE via SSCH command.
Returned by hardware when I/O complete.


Real Address of last CCW executed.
00 00E258 0C00 0000
Difference between number of bytes requested and number read.
Flag '00'
CCW Address '00E258'
Unit Status '0C'
Channel Status '00'
Residual Byte Count '0000'

'80'-'10'	Protection Key
'08'	Unused
'04'	Logout pending
'02'–'01'	Condition code from Start I/O command

'80'	Attention interrupt	'08'	Channel End
'40'	Status Modifier	'04'	Device End
'20'	Control Unit End	'02'	Unit Check
'10'	Busy	'01'	Unit Exception

'80'	Program Controlled Interrupt
'40'	Incorrect Length
'20'	Program Check
'10'	Protection Check
'08'	Channel Data Check
'04'	Channel Control Check
'02'	Interface Control Check
'01'	Chaining Check

Exercise Four: I/O Error

CCB: (Command Control Block) used by application to request Input/Output by VSE)

```
0000 98 00 0C00 0124 00631280 00631298
```

Residual Byte Count	Flag 1	Flag 2	CSW Status	Logical Unit	First CCW in String	Last CCW Executed
‘0000’	‘98’	‘00’	‘0C00’	‘0124’	‘00631280’	‘00631298’

Flag 1:

‘80’	Traffic Bit (Set at Channel End or Device End)
‘40’	End of File
‘20’	Unrecoverable I/O Error
‘10’	Accept Unrecoverable Errors
‘08’	Return Data Checks
‘04’	Post at Device End
‘02’	Return Data or Read Check
‘01’	User Error Routine

Flag 2:

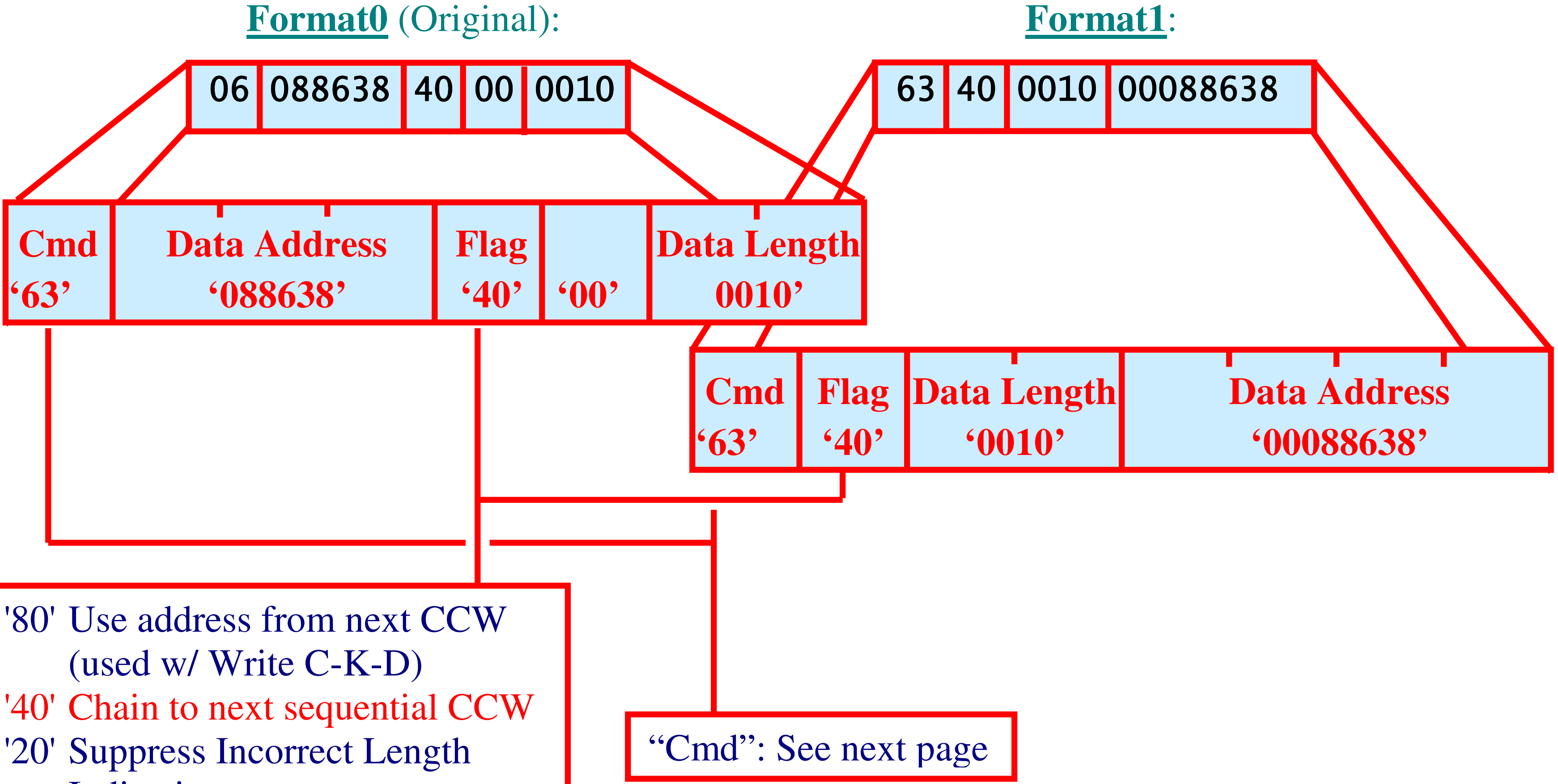
‘80’	Data Check in count area
‘40’	Track Overrun
‘20’	End of Cylinder
‘10’	Data Check
‘08’	No Record Found
‘04’	Retry No-record-found
‘02’	Verify Error
‘01’	Command Chain (retry)

Logical Unit:

x’01xx’: Programmer unit (x’0124’ = SYS036)
Else: ‘0000’ = SYSRDR, ‘0001’ = SYSIPT, ‘0002’ = SYSPCH, ‘0003’ = SYSLST, ‘0004’ = SYSLOG, ‘0005’ = SYSLNK, ‘0006’ = SYSRES etc.

Exercise Four: I/O Error

CCW: (Channel Command Word), specify I/O commands



'80' Use address from next CCW (used w/ Write C-K-D)
'40' Chain to next sequential CCW
'20' Suppress Incorrect Length Indication
'10' Skip (suppress transfer of data to main storage)
'08' Cause Channel Program Controlled Interruption (PCI)
'04' Address refers to an Indirect Data Address Word

"Cmd": See next page

Exercise Four: I/O Error

CCW Command Codes commonly-used for DASD:

Count-Key-Data (CKD):		Extended-Count-Key-Data (ECKD):	
07	Seek	63	Define Extent
1B	Seek Head	47	Locate Record
23	Set Sector		
31	Search ID Equal		
08	TIC		
06	Read Data	06	Read Data
86	Multi-track read	86	Multi-track read
05	Write Data	05	Write Data

CCW Op-codes commonly used for printers:

09	Space one line after printing
0B	Space one line immediately
63	Load FCB
89	Skip to channel one after printing
8B	Skip to channel one immediately

CCW Op-codes commonly used for tape:

01	Write	2F	Backspace file
02	Read forward	3F	Forward space file
07	Rewind	9F	Lead display
0F	Rewind and Unload	DB	Modeset

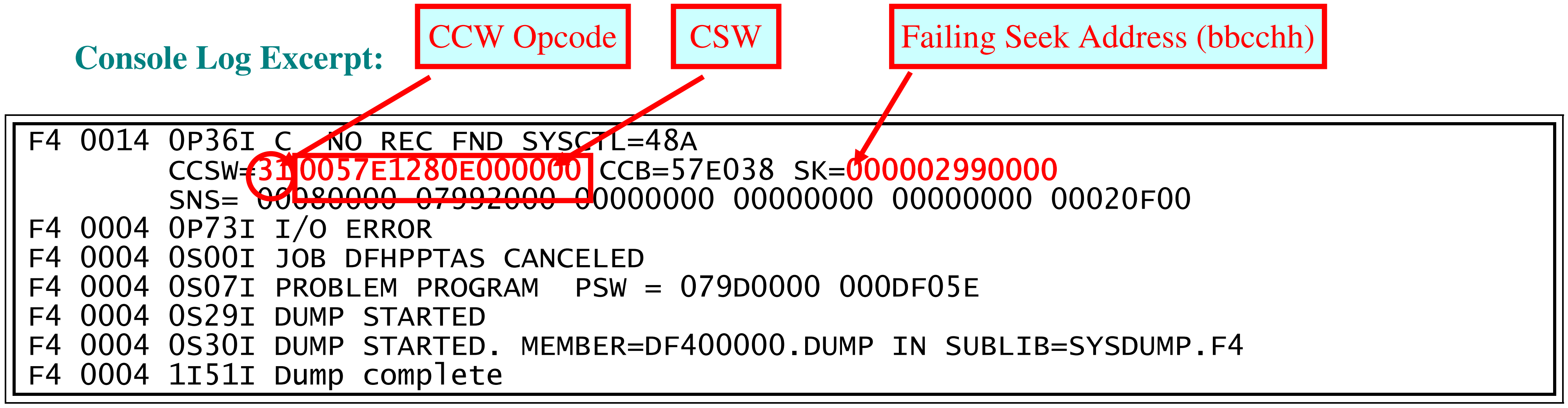
Sense CCW: 04

Exercise Four: I/O Error

Problem Description:

Message 0P36I during compilation of DFHPPT after migration to VSE/ESA 2.1.3.

Console Log Excerpt:



```
F4 0014 0P36I C  NO REC FND SYSCTL=48A
        CCSW=310057E1280E000000 CCB=57E038 SK=000002990000
        SNS= 00080000 07992000 00000000 00000000 00000000 00020F00
F4 0004 0P73I I/O ERROR
F4 0004 0S00I JOB DFHPPTAS CANCELED
F4 0004 0S07I PROBLEM PROGRAM  PSW = 079D0000 000DF05E
F4 0004 0S29I DUMP STARTED
F4 0004 0S30I DUMP STARTED. MEMBER=DF400000.DUMP IN SUBLIB=SYSDUMP.F4
F4 0004 1I51I Dump complete
```

Input/Output errors are documented via a message to the console, 0P01 – 0P68 or 0Exx, and the application is cancelled, unless the application has requested that I/O errors be returned. (CCB+x’02’ = x’10’, x’08’, or x’02’)

1. Check message in _VSE/ESA Messages and Codes, Volume 1_. If the message indicates a critical I/O error, contact hardware support. Many of these kinds of problems are caused by hardware.

Exercise Four: I/O Error

2. The 0Pxx message will identify the failing hardware unit. However, not all hardware errors are logged via EREP. If VSE is running on VM, it passes most error information to VM for logging. “No Record Found” (msg0P36I), “Phase not found” (msg0P62I) or “Wrong Length Record” errors are not logged.
3. Even if a dump is not created (“// OPTION NODUMP”), message 0S07I will include the failing PSW. Ascertain which phase was involved. If the phase was in the SVA, use the “**SHOW**” command to locate an eyecatcher.
4. Errors associated with file management products (LIBR, VSAM, DL/I) are often caused by file corruption. Run the following utilities:
   - LIBR: ($IJBLBR, modules start with INL*)

     Run LIBR TEST against the failing library.
   - VSAM: (modules start with IKQ*)

     Run IDCAMS REPRO against the failing file.

     (modules start with IGG*)

     Run IKQVCHK against the failing catalog.
   - DL/I: (modules start with DLZ*)

     Run DLZURGP0 to unload the database,

     IKQVCHK, to check the catalog

Exercise Four: I/O Error

CCB example: (Define Extent / Locate Record)

```
V00667B80 00001400 000005C7 00667840 01000000 B0 *.......G... ....*
V00667B90 00000000 00000000 00000000 00667BC0 B0 *..............#.*
```

First CCW in string

```
V00667840 63400010 00667BC0 47400010 00667BD0 B0 *. ....#.. ....#.*
V00667850 85002000 011CCA00 03000038 00000000 B0 *................*
V00667860 00000000 00000000 00000000 00000000 B0 *................*
```

Define Extent Block

Extent definition (cchh – cchh):
From Cyl x’13’, hd 0, to Cyl x’1C’ Hd 14

```
V00667BC0 98C02000 00000040 00130000 001C000E B0 *....... ........*
```

Locate Record Block

Record being read (cchhr):
Cyl x’13’, head 1, Record 3

```
V00667BD0 01800001 00130001 00130001 03482000 B0 *................*
V00667BE0 00000000 13000104 00000100 00000000 B0 *................*
```

Exercise Four: I/O Error

```
// JOB DMPANA33  PRINT COMPLETE FORMATTED DUMP                     DATE 10/21/97,CLOCK 14/55/52
// EXEC PROC=DTRINFOA
// ASSGN SYS016,DISK,VOL=SYSWK1,SHR       INFO ANAL MANAGEMENT FILE
1T20I  SYS016 HAS BEEN ASSIGNED TO X'141' (TEMP)
// ASSGN SYS017,DISK,VOL=SYSWK1,SHR       INFO ANAL ROUTINES FILE
1T20I  SYS017 HAS BEEN ASSIGNED TO X'141' (TEMP)
EOP DTRINFOA
// EXEC INFOANA,SIZE=300K

BLN9018I DUMP SYSDUMP.F4.DF400000 SELECTED
          RETURN
   SELECT DUMP VIEWING
          PRINT FORMAT
     .
     .
     .
NAME = CANCLMSG COMPONENT ID = F4       TYPE = TEXT DATA

0P73I I/O ERROR
0S00I JOB DFHPPTAS CANCELED
     .
     .
     .
NAME = GREGS    COMPONENT ID = F4                 DECIMAL DATA

    GP REGS    0-3    00000054 0057E038 0057E000 0057E038
               4-7    FFFFFFFF 000DF0CC 0058F380 00262028
               8-B    0040A190 0040A044 000DBE78 000E002F
               C-F    000DF030 0058F9B0 000DAA90 000DF030

     .
     .
     .
NAME = PSW      COMPONENT ID = F4       TYPE = HEXADECIMAL DATA
       PSW     079D0000 000DF05E
     .
     .
     .
```

Reg1 ⇒ CCB

Failing instruction

Exercise Four: I/O Error

```
NAME=$IJBLBR  COMPONENT ID=SVA      BASE=000BC198
 000BC180                                                                 47F0F046 5BC9D1C2 00 *                                .00.$IJB*
 000BC1A0  D3C2D940 C4C2F6F0 C9C9F0F0 C4F6F1D5 C4E4E3F5 F6F8F660 F0F6F640 4DC35D40 00 *LBR DB60II00D61NDUT5686-066 (C) *
 000BC1C0  C3D6D7E8 D9C9C7C8 E340C9C2 D440C3D6 D9D74B40 F1F9F8F4 6B40F1F9 F9F541F0 00 *COPYRIGHT IBM CORP. 1984, 1995.0*
 000BC1E0  F1601EF0 58F0F000 07FF0000 0000                                           0 *1-.0.00...........F...........?6*
 000BC200  000BE848 000BEB60 000C3100 000C                                           0 *..Y....-.......(.......1.........*
 000BC220  000CA460 000CAD98 000CE890 000C                                          00 *...-.......Y...0...4...?Q..9.....*
 000BC240  000DF030 000DF0D0 000DFB18 000E0E48 000E1C00 000E2008 000E6238 000E7690 00 *..0...0.........................*
 000BC260  000E5DC8 000E80D8 000F2188 000F87E8 000FB3F0 00000000 C9D5D3C3 D7E3C3C8 00 *..)H...Q.......Y...0....INLCPTCH*
 000BC280  40404040 40404040 40404040 40404040 40404040 40404040 40404040 40404040 00 *                                *
 000BC2A0  TO NEXT LINE ADDRESS SAME AS ABOVE
```

Reg12 points here (begin of module)

```
      •
      •
      •
 000DF000  00600                                                      200000 089D0000 00 *.-.......-.......-..............*
 000DF020  00200000 08FFFFFF FF000000 00000000 47F0F01C 17C9D5D3 D7C4D6C9 D6F5F6F8 00 *.................00..INLPDOIO568*
 000DF040  F6F6F1C9 C9F0F8F3 F1F8F4F5 90ECD00C 18CF41B0 CFFF4150 C09C1813 0A009180 00 *661II0831845...........&........*
 000DF060  10024710 C0380A07 1F001FEE 41F00047 0A6B1FFF 58E0D00C 980CD014 07FED7C1 00 *.............0...,...........PA*
 000DF080  E3C3C840 C1D9C5C1 406040C9 D5D3D7C4 D6C9D640 F9F54BF0 F8F3C06A C06CC06E 00 *TCH AREA - INLPDOIO 95.083...%.>*
      •
      •
      •
BLN5014I DATA FROM 00268D78 TO 0026FD77 NOT AVAILABLE OR ALL ZEROS
BLN9012I PRINT FUNCTION COMPLETED
          PRINT 0 END
      •
      •
      •
```

Branch instructions shows Reg12 to be base reg

PSW pointed here

Exercise Four: I/O Error

```
00400000  D3D5D2C5 C4E34040 079D0000 000DF05E 0040A044 000DBE78 000E002F 000DF030 90 *LNKEDT  ......0;. ............0.*
00400020  0058F9B0 000DAA90 000DF030 00000054 0057E038 0057E000 0057E038 FFFFFFFF 90 *..9.......0.....................*
00400040  000DF0CC 0058F380 00262028 0040A190 0000AF67 5EA86815 40404040 40404040 90 *..0...3...... ......;...        *
00400060  40404040 40404040 40404040 40404040 40404040 40404040 C9D1C2D3 C5F14040 90 *                        IJBLE1  *
00400080  F1F5C3F0 40F2F140 C4F6F1E9 C4C8E240 F5F6F8F6 60F0F6F6 404DC35D 40C3D6D7 90 *15C0 21 D61ZDHS 5686-066 (C) COP*
004000A0  E8D9C9C7 C8E340C9 C2D440C3 D6D9D74B 40F1F9F7 F96B40F1 F9F9F600 00000000 90 *YRIGHT IBM CORP. 1979, 1996.....*
004000C0  D540C1E4 E3D64040 00000004 00423304 00423304 C4C6C8D7 D7E3C1E2 07000000 90 *N AUTO  ............DFHPPTAS....*
004000E0  00000000 00000000 00000000 00400078 00428090 00000000 00000000 00004000 90 *............ ...............  .*
00400100  00000000 C4C6C8D7 D7E3C2C1 01000040 00000003 00000000 00000000 00000000 90 *....DFHPPTBA... ................*
```

•

•

•

Reg1 pointed here (CCB)

```
0057E000  C2E4C3C2 0057E110 00000017 00000006 00000400 00000000 00000000 0057E250 90 *BUCB..........................S&*
0057E020  0057E250 0057E250 0057E290 0057E250 0057E1D0 00000000 0000A00C 0E000831 90 *..S&..S&..S...S&................*
0057E040  0057E110 0057E128 00000000 005F0057 E0B00000 00000000 00000000 00000000 90 *................................*
0057E060  00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 90 *................................*
0057E080  TO NEXT LINE ADDRESS SAME AS ABOVE
0057E100  00000000 00000000 00000000 00000000 0757E25E 60000006 2358FB48 60000001 90 *..................S;-.......-...*
0057E120  3157E260 60000005 0857E120 60000001 8657F020 20000400 00000000 00000000 90 *..S--.......-.....0.............*
0057E140  00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 90 *................................*
0057E160  TO NEXT LINE ADDRESS SAME AS ABOVE
0057E240  FFFFFFFF FFFFFFFF FFFFFFFF 0057F420 C2C8C4D9 00000003 0C600000 00310000 90 *..............4.BHDR.....-......*
0057E260  02990000 01000000 00007F26 FFFFFFFF 00265DD8 FFFFFFFF FFFFFFFF FFFFFFFF 90 *.........."......)Q............*
0057E280  FFFFFFFF FFFFFFFF FFFFFFFF 0057F00 C2C8C4D9 00000004 00000000 00000000 90 *..............0.BHDR...........*
```

Exercise Four: I/O Error

Summary:

1. PSW points at DF05E. (INLPDOIO + X'2E'). Last instruction executed was I/O wait (SVC 07). PSW has been backed up 10 bytes:

```
 9180 1002        TM   CCBFLAG,POSTED
 4710 C038        BO   *+6
 0A07             SVC  07
```

2. Register 1 points at the CCB (0057E038) = `0000A00C 0E000831 0057E110 0057E128`
3. CSW Status Bytes ('0E00') indicate Channel End + Device End + Unit Check

   CCB communications bytes (`'A00C'`) indicate Posted + Unrecoverable I/O error + No Record Found.

   I/O was performed to system device x'31' (SYSCTL). Msg0P36I identified SYSCTL as being assigned to device '48A'.

   CCW chain starts at 57E110, and last CCW executed was at 57E120 (57E128 – 8)
4. CCW chain at 57E110:

`0757E25E 60000006`	Seek to cylinder x'299' (665) {bbcc = 00000299}
`2358FB48 60000001`	Set Sector to 04 (not in handout)
`3157E260 60000005`	Search to cylinder x'299', head 0, record 1 {cchhr = 0299000001}
`0857E120 60000001`	TIC back to previous search (keeps channel program active until head is positioned properly)
`8657F020 20000400`	Multi-track read. 1024 (x'400') byte record into buffer at x'57F020'.

5. CCW chain was terminated prematurely at search with "no record found".

Exercise Four: I/O Error

Problem Resolution:

To get around the immediate problem, the library was re-built. Cylinder 665 of device x'48A' was part of the VSE library. However, when VSE/DITTO was used to display this area, it showed all records (including record 1) missing from this track (and subsequent tracks for several cylinders). Turning on SYS DASDFP=YES during IPL, identified a product initializing space outside of its defined extents.

Exercise Four: I/O Error

This page intentionally left blank

CICS TS is not Responding

CICS TS in a Partition Wait State

1. Does CICS respond to External Interrupt?

MSG F2,DATA='CEMT I TA'

```
          Tas(0000025) Tra(IESO)           Sus Tas Pri( 020 )
          Sta(S ) Use(CICSUSER) Rec(X'B59A0E8A80867884') Hty(EKCWAIT )
          Tas(0000072) Tra(TST2) Fac(A001) Sus Ter Pri( 001 )
          Sta(TO) Use(SYSA    ) Rec(X'B59A29ABBC9EB5CB') Hty(ZCIOWAIT)
          Tas(0000081) Tra(TST2) Fac(A002) Sus Ter Pri( 001 )
          Sta(TO) Use(SYSB    ) Rec(X'B59A2AD6DC5C528A') Hty(FCXCWAIT)
          Tas(0000090) Tra(CEMT) Fac(CNSL) Run Ter Pri( 255 )
          Sta(TO) Use(CICSUSER) Rec(X'B59A2B6535A2A387')
     .
     .
     .
```

Yes ... goto page 17 (Transaction Waits)

CICS TS is not Responding

a. Is CICS TS being dispatched? Check CSATODP (CSA+x’50’):

```
LOCATE F2,'AICA
AR+0015   MATCH FOUND AT 0058ADB0
V0058ADB0 C1C9C3C1 1C000000 E2E3D6D9 C1C7C540 B6 *AICA....STORAGE *
R02078DB0
V0058ADC0 00000248 0050E8A0 0050B780 81394060 B6 *.....&Y..&..a. -*
R02078DC0
15 E
AR 0015 1I40I  READY

SHOW F2,58ADB0.70
V0058ADB0 C1C9C3C1 1C000000 E2E3D6D9 C1C7C540 B6 *AICA....STORAGE *
R02078DB0
V0058ADC0 00000248 0050E8A0 0050B780 81394060 B6 *.....&Y..&..a. -*
R02078DC0
V0058ADD0 00000000 02213080 013940D0 00509000 B6 *.......... }.&..*
R02078DD0
V0058ADE0 81393E40 017A0700 0000000B 0194D000 B6 *a.. .:.......m}.*
R02078DE0
V0058ADF0 01966458 019664E2 01966587 8058ADC0 B6 *.o...o.S.o.g...{*
R02078DF0
V0058AE00 01280080 01280080 0010020C 0127E680 B6 *.............W.*
R02078E00
V0058AE10 1307070F 017AB2D0 00000100 018B8200 B6 *.....:.}......b.*
R02078E10

SHOW F2,58ADB0.70
V0058ADB0 C1C9C3C1 1C000000 E2E3D6D9 C1C7C540 B6 *AICA....STORAGE *
R02078DB0
```

CICS TS is not Responding

```
V0058ADC0 00000248 0050E8A0 0050B780 81394060 B6 *.....&Y..&..a. -*
R02078DC0
V0058ADD0 00000000 02213080 013940D0 00509000 B6 *.......... }.&..*
R02078DD0
V0058ADE0 81393E40 017A0700 0000000B 0194D000 B6 *a.. .:.......m}.*
R02078DE0
V0058ADF0 01966458 019664E2 01966587 8058ADC0 B6 *.o...o.S.o.g...{*
R02078DF0
V0058AE00 01280080 01280080 0010020C 0127E680 B6 *..............W.*
R02078E00
V0058AE10 1307112F 017AB2D0 00000100 018B8200 B6 *.....:.}......b.*
R02078E10
```

Yes ... goto 4a (Transaction Wait)

CICS TS is not Responding

Speaker Notes:

You can locate the CSA by searching for the printable string “AICA   STORAGE”. Note, there is no terminating quote in the command. The first quote is a special symbol to notify the LOCATE command that you are searching for printable data (as opposed to hex data). Don’t forget to enter “15 e” to terminate the locate command. Otherwise it will lock the attention routine and reject all further console commands.

The CSA starts at +x’10’ after this eyecatcher. Last dispatched time-of-day (HHMMSST) is a packed field (ends with “F”), and is located at CSA+x’50’. This value changes each time CICS TS goes through task dispatching (Dispatcher Domain). If this does not change, CICS is either in a partition wait, a loop in CICS TS system code, or a transaction has “seized” the system.

The CSA+x’4C’ points at the user area of the current TCA. The first word points at the system area of the TCA, which will be a x’100’ higher address. This System TCA will contain printable data fields for the current transaction, containing the current transaction and transaction status, in the case of an abend.

CICS TS Usage of VSE/ESA Sub-tasks

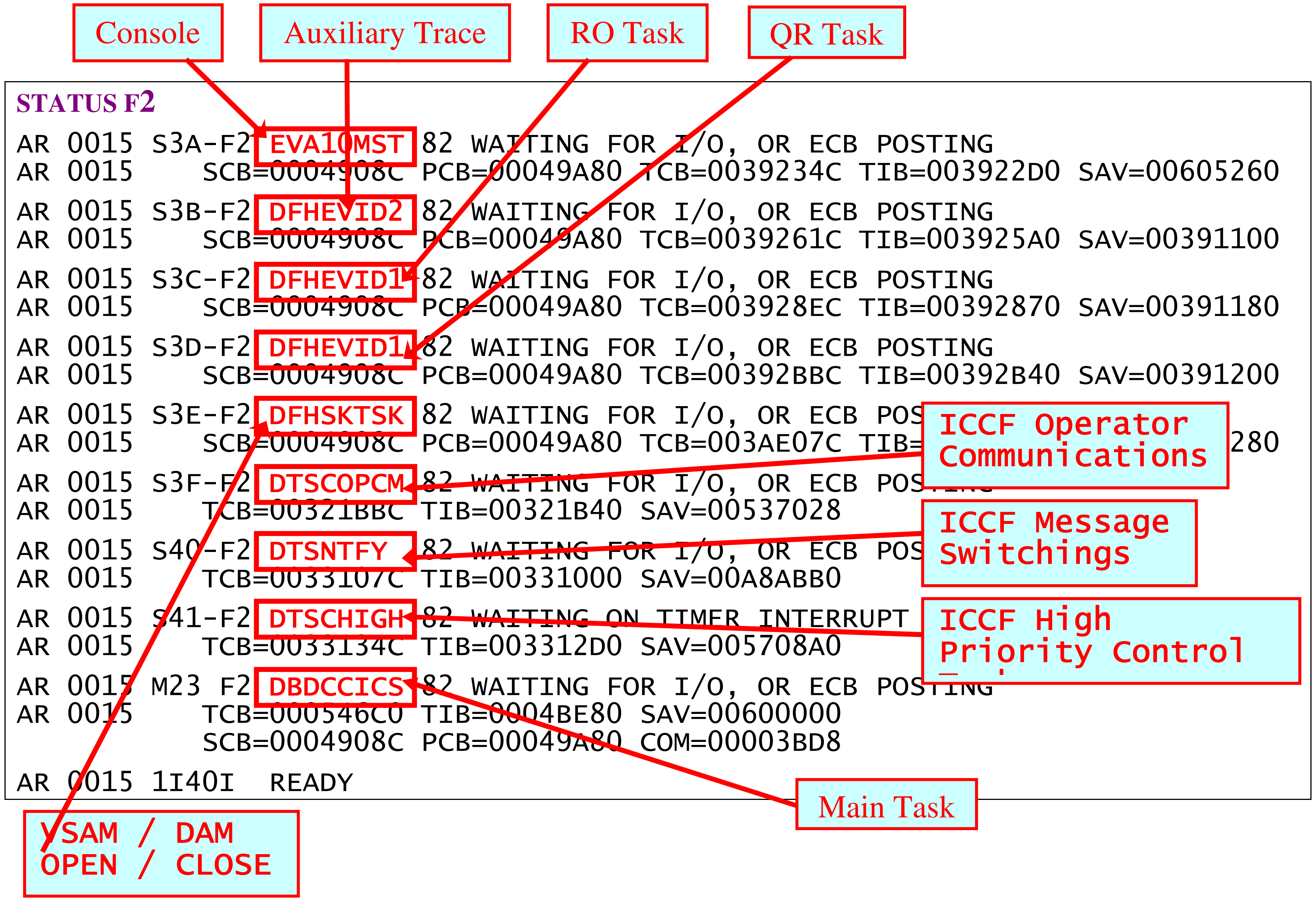
CICS TS uses multiple VSE/ESA subtasks, and the main task is dormant until CICS is terminated. Up to five VSE/ESA tasks are used to run CICS TS user transactions, and have the same name (`DFHEVID1`):

- **Quasi-Reentrant (QR) task:** this runs "normal" task code
- **Resource-Owning (RO) task:** CICS code will switch to this task when a long VSE/ESA wait will occur that would stop other CICS tasks from being dispatched via the QR subtask e.g. OPEN and CLOSE TD files, CICS-managed files such as dump datasets, load a program, journal archive submit, security SVC interface etc.
- **Secondary LU Usage (SZ) Task:** Front End Programming Interface (FEPI)
- **Sockets (SO) Task:** CICS Web Support
- **Sockets Listener (SL) Task:** CICS Web Support

  In the VSE/ESA STATUS command display,
  the order from top is:
  RO, QR, then SZ, SO, and SL, if active

Uh oh, now we're really getting in deep!

CICS TS Usage of VSE/ESA Sub-tasks



```
STATUS F2
AR 0015 S3A-F2 EVA1QMST 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     SCB=0004908C PCB=00049A80 TCB=0039234C TIB=003922D0 SAV=00605260
AR 0015 S3B-F2 DFHEVID2 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     SCB=0004908C PCB=00049A80 TCB=0039261C TIB=003925A0 SAV=00391100
AR 0015 S3C-F2 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     SCB=0004908C PCB=00049A80 TCB=003928EC TIB=00392870 SAV=00391180
AR 0015 S3D-F2 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     SCB=0004908C PCB=00049A80 TCB=00392BBC TIB=00392B40 SAV=00391200
AR 0015 S3E-F2 DFHSKTSK 82 WAITING FOR I/O, OR ECB POS
AR 0015     SCB=0004908C PCB=00049A80 TCB=003AE07C TIB=                  280
AR 0015 S3F-F2 DTSCOPCM 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     TCB=00321BBC TIB=00321B40 SAV=00537028
AR 0015 S40-F2 DTSNTFY  82 WAITING FOR I/O, OR ECB POS
AR 0015     TCB=0033107C TIB=00331000 SAV=00A8ABB0
AR 0015 S41-F2 DTSCHIGH 82 WAITING ON TIMER INTERRUPT
AR 0015     TCB=0033134C TIB=003312D0 SAV=005708A0
AR 0015 M23 F2 DBDCCICS 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     TCB=000546C0 TIB=0004BE80 SAV=00600000
            SCB=0004908C PCB=00049A80 COM=00003BD8

AR 0015 1I40I  READY
```

CICS TS Usage of VSE/ESA Sub-tasks

VSE Sub-task 4 (QR) (DFHEVID1, TASKID 3D) (Normally)

```
AR 0015 S3D-F2 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     SCB=0004908C PCB=00049A80 TCB=00392BBC TIB=00392B40 SAV=00391200
```

SHOW 391200.60

```
AR 0015    DATA   FOUND AT 00391200
V00391200 C4C6C8C5 E5C9C4F1 07BD1000 81F498CC 06 *DFHEVID1....a4q.* R00256400
V00391210 01E2D2689 01E280D4A 00000001B 01E2D268C 06 *.SK..S.M.....SK.* R00256410
V00391220 01F0BAC0D 00000006E 2B8BEC74F 000000010 06 *.0.{............* R00256420
V00391230 FE1D80001 01E2D2682 81F490E03 01E2D0004 06 *.....SK.a4.\.S}.* R00256430
V00391240 01F4A0DF5 000000006 806053B07 01EF60008 06      ..-....-.* R00256440
```

PSW

Base Register

CICS TS Usage of VSE/ESA Sub-tasks

Base Register

SHOW 1F490E0.60

```
V01F490E0 00486ED4 D6C4C8C5 C1C403D7 00F4F1F0 B4 *..>MODHEAD.P.410* R001380E0
V01F490F0 C4C6C8C4 E2C4E2F3 F0F161F2 F061F9F9 B4 *DFHDSDS301/20/99* R001380F0
V01F49100 7CF1F14B F4F10005 81F49B20 D1E5E2C7 B4 *@11.41..a4..JVSG* R00138100
V01F49110 40404040 1E380218 00000002 00001E48 B4 *    ............* R00138110
V01F49120 00000360 00000000 183F4150 3FFF1840 B4 *...-.......&... * R00138120
     •
     •
     •
V01F498C0 41101000 13110700 070 0A84 07000700 B4 *...........d....* R001388C0
V01F498D0 0A01 205 803098EF 80301824 1E269801 B4 *......q.......q.* R001388D0
V01F498E0 22A81FE0 1FF147B0 38104100 00011FE0 B4 *.Y.\.1..........* R001388E0
     •
     •
     •
V01F4AB20 92687001 41000028 50007004 18175890 B4 *k.......&.......* R00139B20
V01F4AB30 D05498F0 90300DEF 94FDD219 58E0D238 B4 *}.q0.....m.K..\K.* R00139B30
V01F4AB40 07FED7C1 E3C3C840 C1D9C5C1 406040C4 B4 *..PATCH AREA - D* R00139B40
V01F4AB50 C6C8C4E2 C4E2F340 F9F94BF0 F2F05A7F B4 *FHDSDS3 99.020!"* R00139B50
V01F4AB60 5A815A83 5A855A87 5A895A8B 5A8D5A8F B4 *!a!c!e!g!i!.!.!.* R00139B60
```

PSW

0A840700 07000A01:	SIMSVC instruction format.
0A84 (SVC 132)	Calls VSE/ESA OS/390 emulation.
07000700:	Filler
0A01:	OS/390 SVC for Execute Channel program (EXCP)

CICS TS Usage of VSE/ESA Sub-tasks

Normally, the save area for the quasi-reentrant task will be pointing as shown on previous page, but …

```
AR 0015 S3D-F2 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING
AR 0015     SCB=0004908C PCB=00049A80 TCB=00392BBC TIB=00392B40 SAV=00391200
```

SHOW 00391200.60

```
V00391200 C4C6C8C5 E5C9C4F1 07BD0000 801D23E0 06 *DFHEVID1.......\* R00256200
V00391210 00667BE0900628310A00667850B001D2318C06 *..#\..c....&....* R00256210
V00391220 011FAA18D801D39A6E001D457FF801D408E006 *.......w...".. .* R00256220
V00391230 00667B801000000002 01E13D843 00667B80406 *...............* R00256230
V00391240 000004205011FA198600628418 7011FA198806 *...............q* R00256240
V00391250 0000B1FF 003D54EF 40404040 40404040 06 *........        * R00256250
AR 0015 1I40I READY

V001D2310 00000000 00000000 47F0C018 C9D2D8C9 04 *.........0{.IKQI*
R005EA310
V001D2320 D6C44040 F4F5C340 C4E8F4F4 F7F7F800 04 *OD  45C DY44778.*
R005EA320
V001D2330 4560C8A0 5860D108 58106030 12114780 04 *.-H..-J...-.....*
R005EA330
V001D2340 C0D49110 10024710 C0CA1B55 BF571009 04 *{Mj.....{.......*
R005EA340
V001D2350 951B5000 4770C044 92075000 58501020 04 *n.&...{.k.&..&..*
R005EA350
V001D2360 4A55003A 4B50C900 947F5001 9238500B 04 *¢....&I.m"&.k.&.*
R005EA360
      •
      •
      •
V001D23C0 0A4112FF 4780C0B4 1B000A06 5010B084 04 *......{.....&..d*
R005EA3C0
```

Base Reg (R12)

CCB Ptr (R1)

CICS TS Usage of VSE/ESA Sub-tasks

```
V001D23D0 181518E6 47F0C086 05EF18E6 47F0C0CA 04 *...W.0{f...W.0{.*
R005EA3D0
V001D23E0 91801002 4710C0D2 0A07C02A 4560C804 04 *j.....{K..{..-H.*
R004A23E0
V001D23F0 58C0D14C 07FE4BC0 C826180E 5880D108 04 *.{J<...{H.....J.*
R004A23F0
```

CICS TS Usage of VSE/ESA Sub-tasks

Speaker Notes

This task is waiting in VSE/VSAM I/O routine IKQIOD. Register one points at a CCB (see following) which indicates that VSE/VSAM is currently reading from cylinder 19, head 1, record 3 on the device assigned to SYS199 (x’05C7’ in CCB+x’0A’,
drop off the ‘05’ and convert x’C7’ to decimal).
The actual device can be identified by releasing a pause job and issuing the following command:

```
        LISTIO F2
```

In this case, the customer was experiencing a corrupt file index, which was causing VSE/VSAM to loop reading the same index records over and over. See “Design and Tuning of VSE/VSAM” to identify which cluster was corrupted.

Reg1 (CCB):

```
V00667B80 00001400 000005C7 00667840 01000000 B0 *.......G... ....*
V00667B90 00000000 00000000 00000000 00667BC0 B0 *..............#.*
```

First CCW in string

```
V00667840 63400010 00667BC0 47400010 00667BD0 B0 *. ....#.. ....#.*
V00667850 85002000 011CCA00 03000038 00000000 B0 *................*
V00667860 00000000 00000000 00000000 00000000 B0 *................*
```

Define Extent Block

Extent definition (cchh – cchh):
From Cyl x’13’, hd 0, to Cyl x’1C’ Hd 14

```
V00667BC0 98C02000 00000040 00130000 001C000E B0 *....... ........*
```

Locate Record Block

Record being read (cchhr):
Cyl x’13’, head 1, Record 3

```
V00667BD0 01800001 00130001 00130001 03482000 B0 *................*
V00667BE0 00000000 13000104 00000100 00000000 B0 *................*
```

CICS TS Transaction Waits (Hangs)

CICS TS transaction(s) in a wait state

For Transaction status, use **CEMT**:

Task waiting on exclusive access to a VSAM record. File: KSDS

CEMT I TA

```
STATUS:  RESULTS - OVERTYPE TO MODIFY
  Tas(0000025) Tra(IESO)            Sus Tas Pri( 020 )
     Sta(S ) Use(CICSUSER) Rec(X'B59A0E8A80867884') Hty(EKCWAIT )
  Tas(0000072) Tra(TST2) Fac(A001) Sus Ter Pri( 001 )
     Sta(TO) Use(SYSA    ) Rec(X'B59A29ABBC9EB5CB') Hty(ZCIOWAIT)
? Tas(0000081) Tra(TST2)       Fac(A002) Sus Ter Pri( 001 )
     Sta(TO) Use(SYSB    ) Rec(X'B59A2AD6DC5C528A') Hty(FCXCWAIT)
  Tas(0000089) Tra(CEMT) Fac(A003) Run Ter Pri( 255 )
     Sta(TO) Use(OPER    ) Rec(X'B59A2B0E2DCE7EC2')

RESULT - OVERTYPE TO MODIFY
   Task(0000081)
   Tranid(TST2)
   Facility(A002)
   Runstatus(Suspended)

   Userid(SYSB)
   Recunitid(X'B59A2AD6DC5C528A')
   Htype(FCXCWAIT)
   Hvalue(KSDS)
```

CICS TS Transaction Waits (Hangs)

Or, from the VSE console:

MSG F2,DATA='CEMT I TA'

```
         Tas(0000025) Tra(IESO)           Sus Tas Pri( 020 )
         Sta(S ) Use(CICSUSER) Rec(X'B59A0E8A80867884') Hty(EKCWAIT )
         Tas(0000072) Tra(TST2) Fac(A001) Sus Ter Pri( 001 )
         Sta(TO) Use(SYSA    ) Rec(X'B59A29ABBC9EB5CB') Hty(ZCIOWAIT)
         Tas(0000081) Tra(TST2) Fac(A002) Sus Ter Pri( 001 )
         Sta(TO) Use(SYSB    ) Rec(X'B59A2AD6DC5C528A') Hty(FCXCWAIT)
         Tas(0000090) Tra(CEMT) Fac(CNSL) Run Ter Pri( 255 )
         Sta(TO) Use(CICSUSER) Rec(X'B59A2B6535A2A387')
```

"If your task is waiting on resource type FCXCWAIT, it means that it cannot get exclusive control of a VSAM control interval at the present time. Another task already has shared or exclusive control of the control interval, so your task is suspended pending the release of that control interval. An exclusive control wait on resource type FCXCWAIT occurs within CICS, unlike the similar wait on FCIOWAIT, which occurs within VSAM." "CICS TS Problem Determination Guide"

CICS TS Transaction Waits (Hangs)

To diagnose Transaction Wait Problems:

1. Format System Dump:Use **DFHPD410 DEF=0,KE=3,AP=3,DS=1,FCP=3,TR=1 . WAIT_OLDC**".
2. Locate the task in the Kernel Domain KE_Task Summary:
3. Locate TCA:
4. Locate User Savearea:
5. Locate the System Exec Interface Block (EIB):

If transaction is waiting on VSE/VSAM File access:

6. Identify transaction holding the resource we are requesting:
7. Locate FRTE for failing transaction:
8. Locate RPL Message Area:
9. If this transaction is waiting because there are no more strings available:
10. Locate program storage:

CICS TS Transaction Waits (Hangs)

Speaker Notes: (for following page):

1. Format System Dump:Use **DFHPD410 DEF=0,KE=3,AP=3,DS=1,FCP=3,TR=1 .** Locate waiting task in the Dispatcher Domain Summary, either by the "Resource Type / Name" or via "C" in the "W" column. Most transactions waiting on ECBs are handled by CICS TS as "**C=WAIT_OLDC**".
2. Locate the task in the Kernel Domain KE_Task Summary: Use the KE_Task (address). Note the Transaction Number and TCA address. The Transaction Number will match the "**CEMT I TA**" display.
3. Locate TCA: Either by using the address or by searching for "**TCA.NNNNN**", where "nnnnn" is the Transaction Number.
4. Locate User Savearea: Pointer at x'40' into System TCA. Since this time, this is a 31-bit address, it will be located in a EUDSA storage area. Search for "**USER31.NNNNN**", then locate the address in the right-hand column.

At this point we know why the transaction is waiting (which we already knew from the "CEMT I TA"), and we know the address in the user program (but not the statement number). Before we go after the user program storage, let's check to see what command the failing transaction issued, and also extract the transaction history from the trace file.

5. Locate the System Exec Interface Block (EIB): Check EIBFN (displacement x'1B'). In this case, x'0602' indicates "Read File", which we kinda figured already. For File Access Request, displacement x'23' contains the dataset name (EIBDS).

If transaction is waiting on VSE/VSAM File access:

6. Identify transaction holding the resource we are requesting: All long-running requests against a file are tracked via File Request Elements (FRTEs), which are chained from the File Control Table Element (FCTE).
7. Locate FRTE for failing transaction: FRTEs are not formatted in the same section as the FCTE. To locate the FRTE for the failing transaction, search on "**FRTE**", and scan for transaction id. Otherwise, search for the FRTE matching the address in the FCTE +x'**54**'.. FRTE + x'30' points at the VSWA, which contains the RPL starting at displacement x'**08**'. The VSWAs are formatted, but not associated with the transaction Search for "**VSWA**" and compare the address from the FRTE.
8. Locate RPL Message Area: When VSE/VSAM refuses a request because the resource is already owned (exclusive control conflict), it returns an area which points to the transaction owning this request. VSWA+x'**3C**' points to this RPL Message Area. This is translated by CICS TS into a pointer to the VSWA for which I am waiting (VSWA +x'80').
9. If this transaction is waiting because there are no more strings available:
   FCTE +x'**7E**' (Number of tasks waiting on strings) will be non-zero, and the VSAM RPL Feedback will be zero. In addition, the VSWA +x'**80**' will be zero, indicating that this request is not waiting on another specific request.
10. Locate program storage: The program which is waiting is not formatted by DFHPD410, and must be printed using INFOANA. See example on page **Error! Bookmark not defined.**. You can then locate the source statement using the procedure on page **Error! Bookmark not defined.**.

CICS TS Transaction Waits (Hangs)

1. Locate waiting task in the Dispatcher Domain Summary:

```
DEF=0,KE=3,AP=3,DS=1,FCP=3,TR=1
```

```
==DS: DISPATCHER DOMAIN - SUMMARY
DS_TOKEN KE_TASK  T S F P TT RESOURCE RESOURCE_NAME   W TIME OF      TIMEOUT      DTA      AD ATTACHER M SUSPAREA
                             TYPE                       SUSPEND      DUE          (DSTSK)     TOKEN
.
00000003 035FAC80 S S N N -  ZC       DFHZNAC1        S 14:53:27.050      -       03501080 XM 01C06700 Q 03501080
00020003 035FA900 S S N N -  TIEXPIRY DS_NUDGE        S 15:46:17.114      -       03501180 TI 00470003 Q 03500710

00940001 035FA580 S S N N IN SMSYSTEM                 S 15:50:37.161 15:55:37.161 01BDFA80 SM 00000002 Q 03500680
01800001 0215E080 N S P N -  OPEN_ANY DFHPSPIO        W 12:46:16.816      -       01BE8080 XM 01C06300 Q 021C90F9
01840001 02151780 N S N N -  EKCWAIT                  W 12:46:18.168      -       01BE8280 XM 01C07700 Q 00614511
01860035 02151400 N R                                                             01BE8380 XM 01C07100 Q
01880017 02151B00 N S P N -  ZCIOWAIT DFHZARQ1        S 14:47:43.119      -       01BE8480 XM 01C06B00 Q 01BE8480
018C0010 02151080 N S P N -  FCXCWAIT KSDS            C 14:52:55.130      -       01BE8680 XM 01C07300 Q 005A26B4

     T  = TYPE OF TASK           S=SYSTEM  N=NON-SYSTEM
     S  = STATE OF TASK          D=DISPATCHABLE  S=SUSPENDED  R=RUNNING  E=RESUMED EARLY
     F  = PURGEABILITY FLAG      P=PURGEABLE  N=NOT PURGEABLE
     P  = PURGE STATUS           N=NO PURGE  X=PURGED  P=PURGE PENDING
     TT = TIMEOUT TYPE           IN=INTERVAL  DD=DEADLOCK DELAYED  DI=DEADLOCK IMMEDIATE
     W  = WAIT/SUSPEND TYPE      X=WAIT_EXTERNAL  S=SUSPEND  C=WAIT_OLDC  W=WAIT_OLDW
     DTA= DISPATCHER TASK AREA
     AD = ATTACHING DOMAIN
     M  = TASK MODE              Q=QUASI-REENTRANT  R=RESOURCE OWNING  S=FEPI OWNING  O=SOCKET OWNING  L=LISTENER
```

CICS TS Transaction Waits (Hangs)

2. Locate task in the Kernel Domain Task Summary:

```
==KE: Kernel Domain KE_TASK Summary

KE_NUM KE_TASK  STATUS       TCA_ADDR TRAN_# TRANSID DS_TASK  KE_KTCB  ERROR
0001   035F3880 KTCB Step    00000000                00000000 035D70B0
0002   035F3500 KTCB QR      00000000                03505000 035D9020
   •
   •
   •
001F   02151080 Not Running  00597080 00081  TST2    01BE8680 035D9020
0020   02151400 ***Running** 00597680 00100  CEMT    01BE8380 035D9020
0021   02151780 Not Running  00595080 00025  IESO    01BE8280 035D9020
0022   02151B00 Not Running  00595680 00072  TST2    01BE8480 035D9020
```

CICS TS Transaction Waits (Hangs)

3. Locate TCA:

```
TCA.00081 00597080 Task Control Area (User Area)

 0000   00597180 00000001 021E9618 00589648  02153990 00000000 00000000 00000000  *..........o...o.................*
 0020   00000000 0000081C 00000000 00000000  00000000 81F85A1E 021F1150 0000012C  *.....................a8!....&....*
 0040   00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
 0060   00000081 01004000 00000040 021E9618  00000000 00000000 00000000 00000000  *...a.. .... ..o.................*
 0080   00000000 021E9618 00000000 00000000  00000000 00000000 00000000 00000000  *......o.........................*
 00A0   00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
 00C0   00000000 00000000 00000000 00000000  00000000 00000000 00000000 80589DC0  *................................*
 00E0   00000000 00000000 00000000 0053A000  00000000 00000000 00000000 02213AB8  *................................*

SYS_TCA.00081 00597180 Task Control Area (System Area)

 0000   00000000 00000000 00000000 00000000  0000081C 01BF050C 0000004A 00000000  *...........................›....*
 0020   00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
 0040   0221A588 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *..vh............................*
 0060   00000000 00000000 00C00000 00000000  00000000 00000000 80000000 00000000  *................................*
 0080   00000000 005974E4 00000000 00000000  00597388 022178C0 006C2128 006C23E8  *.......U...........h.....%...%.Y*
 00A0   00000000 80589080 00000000 00000000  E3E2E3F2 021E9618 00000000 00000000  *................TST2..o.........*
 00C0   00000000 E3E2E3F2 00000000 00000080  00000000 00000000 00000000 00000000  *....TST2........................*
 00E0   00000000 00000000 00000000 00000000  B59A2AD6 DC5C528A 00000000 00000000  *...................O.*..........*
 0100   00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
 0120   00000000 00000000 00000000 00000000  80589400 00000000 00000000 0059756C  *..................m...........%*
```

CICS TS Transaction Waits (Hangs)

4. Locate User Savearea:

```
USER31.00081 02213AF0 USER storage above 16MB

    0000   E4F0F0F0 F0F0F8F1 00000000 00000000  00000000 00000000 00000000 00000000  *U0000081............
02213AF0
    0020   00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000
*......................02213B10
    0040   00000000 00000000 00000000 00000000  022154F0 00000000 00000000 00000000
*................0......02213B30
    0060   00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000
*......................02213B50
      •
      •
      •
    6A80   00000000 00000000 00000000 00000000  00000000 00000000 00108001 02219418
*......................0221A570

    6AA0   00000000 8240268CE 00000000F 0221A6CC0  0221A7401 0221AA002 0221AAA83 000000014  *....b ........w...x
...0221A590

    6AC0   000000005 00614BBC6 000000007 006C20D08  0221A7D89 0240214CA 0240240CB 022178C0C  *...../....%....xQ. .<.
0221A5B0

    6AE0   F3E3C7E3 02219828 03000000 61030220  0221A250 006151FC 0221A758 00000000
*3TGT..q...../...s&./...0221A5D0
```

5. Locate System Exec Interface Block (EIB):

```
SYSEIB.00081 0059748C System EXEC Interface Block

 -0008                                              5CE2E8E2 C5C9C240  *                       *SYSEIB
*
```

CICS TS Transaction Waits (Hangs)

```
  0000   0145252F 0101087F E3E2E3F2 0000081C   C1F0F0F2 00000000 00007D06 02000000
*......."TST2....A002......'.....*
  0020   000000D2 E2C4E240 40404000 00000000   000000D2 E2C4E240 40404000 00000000   *...KSDS      ........KSDS
.....*
  0040   00000000 00000000 00000000 00000000   00000000 00                            *....................
*
```

EIBFN
'0602' = Read

EIBDS
'KSDS'

EIBTR

CICS TS Transaction Waits (Hangs)

6. Locate File Control Table Entry:

Maximum number of strings

Maximum number of update strings

FRTE Chain Pointer

```
FCTE.KSDS 0218FA80 FCT ENTRY

  0000  D2E2C4E2 40404040 0219A160 00000000  0000000C 00EABA0A 80004400 10000000  *KSDS     ...-
....................*
  0020  40400000 00020002 00000000 00000000  00000000 00000000 00000000 00000004  *
............................*
  0040  00000006 00000002 00000000 B59A29AF  41F72ACB 0219D030 0219B670 0219B670
*...............7..............*
  0060  02184204 00000000 00000000 00000000  00000000 00008044 00000000 0001 0000
*..............................*
  0080  000A 0008 00000000 00000000 00000000  00C80000 000B000A 005B4170 40000000  *...............H.......$..
...*
  00A0  00000000 00000000 C9D1E2E8 E2C3E300  40404040 40404040 00000000 00000000  *........IJSYSCT.
........*
  00C0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000
*..............................*
  00E0  00000000 00000000 0000                                                    *...
*
```

Number of active strings

Number of transactions waiting on strings.

ACB.KSDS 005B4170 VSAM ACB (FCTE +x'**98**' points to ACB)

```
  0000  A040004C 00D7F718 001C45A8 0000000B  000ADA20 28110A00 00000000 D2E2C4E2  *.
.<.P7....y................KSDS*
  0020  40404040 00000000 00000000 01B29400  005A21AC 00000000 0A080000 00000000  *
..........m..!..............*
```

CICS TS Transaction Waits (Hangs)

```
 0040   00000000 01000000 00000000                                            *............
*
```

If transaction is waiting on FCTE strings, FCTE +x’7E’ will be non-zero, and the VSAM RPL Feedback will be zero.
In addition, the VSWA +x’80’ will be zero, indicating that this request is not waiting on another specific request.
(See next page.)

CICS TS Transaction Waits (Hangs)

7. Locate FRTE and VSWA:

```
FC FRAB AND FRTE CHAIN FOR TRANSACTION NUMBER: 0000081
DFHFRAB 0219C060 FC FRAB STORAGE
  0000   0219D030 005A2630 00597080 00000000   00000000                                    *.....!..............
*


DFHFRTPS 0219D030 FC FRTE STORAGE
  0000   00000000 00000000 0219D080 00000000   03000000 00000000 00000000 00000000
*................................*
  0020   00000000 0219C060 0218FA80 0219B670   005A2630 00000000 000
.........!..............*
  0040   00000000 00000000 00000000                                                          ............
*



VSWA 005A2630 VSAM WORK AREA
  0000   8F0000E4 00000000 0011003C 00000000   0221AA00 0221A7D8 000000C8 000000C8
*...U..................xQ...H...H*
  0020   005B4170 021C000A 99C00000 00080014   00000000 00000000 00000000 0221C778
*.$......r....................G.*
  0040   00800000 00000000 0218FA80 005A26EC   005A2700 00140010 00000000 00000000
*.............!...!..............*
  0060   00000000 00000000 00000000 00000000   005A2540 00000000 00000000 00000000   *................!.
............*
  0080   005A2540 00900000 0000001C 00000000   00000000 00597080 0219D030 00008000   *.!.
...........................*
```

VSAM RPL Feedback
x'**080014**' = Exclusive control Conflict

CICS TS Transaction Waits (Hangs)

```
  00A0   00000000 00000000 00000000 00000000  00000000 00000000 00000000 0219B670
*................................
  00C0   00000000 00000000 0000                00000 00000000 00000000
*...................?.........
  00E0   00000000                                                              *....
*
```

Pointer to VSWA / RPL we are waiting for

Pointer to RPL Message Area

RPL Message Area:

USER31.00081 0221C770 USER storage above 16MB

```
  0000   E4F0F0F0 F0F0F8F1 D9D7D3C1 C4C4D940  7A404040 005A2548 40404040 40404040  *U0000081RPLADDR :    .!..
*
  0020   40404040 40404040 E3C1E2D2 D5E4D440  7A404040 00072C40 40404040 40404040  *        TASKNUM :    ...
*
  0040   40404040 40404040 E3D9C1D5 E2C9C440  7A404040 E3E2E3F2 40404040 40404040  *        TRANSID :    TST2
*
  0060   40404040 40404040 C6C3E3D5 C1D4C540  7A404040 D2E2C4E2 40404040 40404040  *        FCTNAME :    KSDS
*
  0080   40404040 40404040 E4F0F0F0 F0F0F8F1                                       *        U0000081
*
```

Pointer to VSWA / RPL we are waiting for

This transaction holds the resource we are waiting for

CEMT I TA

```
STATUS:   RESULTS - OVERTYPE TO MODIFY
  Tas(0000025) Tra(IESO)            Sus Tas Pri( 020 )
```

CICS TS Transaction Waits (Hangs)

```
     Sta(S ) Use(CICSUSER) Rec(X'B59A0E8A80867884') Hty(EKCWAIT )

  Tas(0000072) Tra(TST2) Fac(A001) Sus Ter Pri( 001 )
     Sta(TO) Use(SYSA    ) Rec(X'B59A29ABBC9EB5CB') Hty(ZCIOWAIT)

? Tas(0000081) Tra(TST2)       Fac(A002) Sus Ter Pri( 001 )
     Sta(TO) Use(SYSB    ) Rec(X'B59A2AD6DC5C528A') Hty(FCXCWAIT)

  Tas(0000089) Tra(CEMT) Fac(A003) Run Ter Pri( 255 )
     Sta(TO) Use(OPER    ) Rec(X'B59A2B0E2DCE7EC2')
```

CICS TS Transaction Waits (Hangs)

Summary:

1. User complained about a transaction not responding. CEMT Inquire Tasks showed transaction in FCXCWAIT status, indicating it was waiting for File Control resource.

2. We took a CEMT SNAP dump, and formatted it. According to the Dispatcher Domain Summary, the hung transaction was Kernel Task x'**02151080**'. Moving to the Kernel Domain Task Summary, the transaction id is found to be 0081, which matches the initial CEMT Inquire Tasks display.

3. Located the TCA for this task, then the User Savearea. Reg 14 pointed at the last **EXEC CICS** call.

4. The System EIB indicates the last **EXEC CICS** call was a Read Record to VSAM file KSDS.

5. The FCTE for KSDS did not show any transactions waiting on strings.

6. The FRTE for this transaction pointed at a VSWA, which showed that the File I/O request had been denied by VSAM because of an exclusive control conflict. The resource is currently owned by transaction 0072.

7. The trace table showed transaction 81 as waiting on a Get-for-Update on a VSAM record. Transaction 72 had previously read a record in update mode, then started a conversation with a terminal, which had not yet completed. This is a violation of VSAM file access protocol.

Process Stand-alone Dump Tape using INFOANA

Following JCL can be used to define a new SYSDUMP dataset in non-VSE/VSAM space:

```
// JOB DEFINE SYSDUMP AND ASSOCIATED WORK FILES
/* NEW DUMP LABELS
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
// ASSGN SYS010,DISK,TEMP,VOL=vvvvvv,SHR
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0
// EXTENT SYS010,,1,0,nn,mmm
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
/* END OF DUMP LABELS
// EXEC LIBR
   DEFINE L=SYSDUMP
   DEFINE S=SYSDUMP.DYN
   DEFINE S=SYSDUMP.BG
   DEFINE S=SYSDUMP.F1
   DEFINE S=SYSDUMP.F2
/*
// EXEC INFOANA,SIZE=300K
   SELECT DUMP MANAGEMENT
   UTILITY
   RETURN
   SELECT END
/*
```

Process Stand-alone Dump Tape using INFOANA

```
// UPSI 1
// LIBDEF *,SEARCH=(PRD1.BASE)
// EXEC DITTO
$$DITTO CSQ FILEOUT=BLNXTRN,BLKFACTOR=1
ANEXIT IJBXDBUG ANALYZE STANDALONE DUMP ROUTINE
ANEXIT IJBXCSMG ANALYZE CONSOLE BUFFER
ANEXIT IJBXSDA  SDAID BUFFER FORMATTING ROUTINE
ANEXIT DFHPDAP  CICS/VSE (COEXISTANCE) ANALYZER
ANEXIT DFHPD410 CICS DUMP ANALYZER TS 1.1
/*
/&
```

If BLNDMF file gets out of sequence with the VSEDUMP library, INFOANA may report that the library is full, but in fact there are no dumps in the library at all. The following, quick, job will often resynchronize the library, allowing you to continue to use it. If this does not work, the library is probably corrupted, and will need to be re-built.

```
// JOB INIT SYSDUMP MANAGEMENT FILE
/* NEW DUMP LABELS
... SYSDUMP, BLNDMF and BLNXRTN DLBL / EXTENT / ASSGN ...
/* END OF DUMP LABELS
// EXEC INFOANA,SIZE=300K
   SELECT DUMP MANAGEMENT
   UTILITY
   RETURN
   SELECT END
/*
/&
```

Process Stand-alone Dump Tape using INFOANA

Scan the stand-alone dump tape (and identify the dumps to be onloaded)

```
R  RDR,PAUSEBG
BG 0000 // JOB PAUSEBG
BG-0000 // PAUSE
0 // EXEC DOSVSDMP
BG 0000 4G01D SELECT ONE OF THE FOLLOWING FUNCTIONS:

   1    CREATE STAND-ALONE DUMP PROGRAM
   2    SCAN DUMP TAPE/DISK
   3    PRINT DUMP TAPE/DISK
   4    PRINT SDAID TAPE
   5    PRINT IPL DIAGNOSTICS
   R    END DOSVSDMP PROCESSING
0 2
BG 0000 4G04D SPECIFY ADDRESS OF DUMP DEVICE (CUU OR SYSNNN)
BG-0000 OC66D READY

0 7A8
```

Process Stand-alone Dump Tape using INFOANA

SYSLST Output:

```
PRINTOUT OF VSE DUMP TAPE
DIRECTORY OF VSE DUMP TAPE
DUMP FILE  DUMP TYPE  NAME      DATE      DATA DUMPED
---------  ---------  --------  --------  --------------------
   001                              DOES NOT CONTAIN DUMP DATA
   002                              DOES NOT CONTAIN DUMP DATA
   003     SADUMP                         SUPERVISOR+SVA
   004     SADUMP               99/04/14  PMRAS-R
   005     SADUMP               99/04/14  PMRAS-00
   006     SADUMP     T77VDA    99/04/14  Z1-PARTITION
   007     SADUMP     T77VEA    99/04/14  Z2-PARTITION
   009     SADUMP     VSECVTIE  99/04/14  F3-PARTITION
   010     SADUMP     POWSTART  99/04/14  F1-PARTITION
END OF DUMP
```

Process Stand-alone Dump Tape using INFOANA

Load file three (Supervisor and SVA) from a stand-alone dump tape.

```
// JOB INFOANAL ONLOAD S/A DUMP FROM TAPE
// ASSGN SYS002,181
// MTC REW,SYS002
/* NEW DUMP LABELS
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
// ASSGN SYS010,DISK,TEMP,VOL=vvvvvv,SHR
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0
// EXTENT SYS010,,1,0,nn,mmm
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
/* END OF DUMP LABELS
// EXEC INFOANA,SIZE=300K
  SELECT DUMP MANAGEMENT            *** DELETE PREVIOUS DUMPS ***
    DUMP NAME SYSDUMP.F2.SUPSVA
    DELETE
    RETURN
  DUMP NAME SYSDUMP.F2.SUPSVA     *** LOAD NEW DUMP ***
  SELECT DUMP ONLOAD
     VOLID 111111 SYS002
     FILE 3 LAST
     RETURN
  SELECT END
/*
/&
```

Process Stand-alone Dump Tape using INFOANA

Create a list of all dumps currently in the VSEDUMP Library.

```
// JOB INFOANAL LIST ALL DUMPS IN SYSDUMP LIBRARY
/* NEW DUMP LABELS
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
// ASSGN SYS010,DISK,TEMP,VOL=vvvvvv,SHR
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0
// EXTENT SYS010,,1,0,nn,mmm
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
/* END OF DUMP LABELS
// EXEC INFOANA,SIZE=300K
  SELECT DUMP MANAGEMENT            *** LIST ALL DUMPS IN LIBRARY ***
    PRINT DATA
    RETURN
  SELECT END
/*
/&
```

Process Stand-alone Dump Tape using INFOANA

Print Formatted analysis of dump:

```
// JOB INFOANAL ANALYZE S/A DUMP
/* NEW DUMP LABELS
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
// ASSGN SYS010,DISK,TEMP,VOL=vvvvvv,SHR
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0
// EXTENT SYS010,,1,0,nn,mmm
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
/* END OF DUMP LABELS
// EXEC INFOANA,SIZE=300K
   SELECT DUMP MANAGEMENT
     DUMP NAME SYSDUMP.F2.P9141003
          RETURN
   SELECT DUMP VIEWING
          CALL IJBXDBUG
          CALL IJBXSDA
          CALL IJBXCSMG
          RETURN
   SELECT DUMP VIEWING
     PRINT FORMAT
   SELECT END
/*
/&
```

Process Stand-alone Dump Tape using INFOANA

Format CICS TS System Dump, or CICS TS partition from Stand-alone dump:

```
// JOB DFHPD410 FORMAT CICS TS SYSTEM DUMP
/* NEW DUMP LABELS
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
// ASSGN SYS010,DISK,TEMP,VOL=vvvvvv,SHR
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0
// EXTENT SYS010,,1,0,nn,mmm
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
/* END OF DUMP LABELS
// LIBDEF *,SEARCH=(PRD1.BASE)
// EXEC INFOANA,SIZE=300K
  DUMP NAME SYSDUMP.F2.F2DUMP
  SELECT DUMP SYMPTOMS
    PRINT DATA
    RETURN
  SELECT DUMP VIEWING
    CALL DFHPD410 DATA DEF=0,KE=3,AP=3,DS=3,TR=1
    RETURN
  SELECT END
/*
/&
```

Process Stand-alone Dump Tape using INFOANA

Off-load dump to tape to send to VSE/ESA Level2, or for off-line archiving.

```
// JOB OFFLOAD DUMPS FROM VSEDUMP LIBRARY
// ASSGN SYS009,182
// MTC REW,182
/* NEW DUMP LABELS
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
// ASSGN SYS010,DISK,TEMP,VOL=vvvvvv,SHR
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0
// EXTENT SYS010,,1,0,nn,mmm
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
/* END OF DUMP LABELS
// EXEC INFOANA,SIZE=300K
  DUMP NAME SYSDUMP.F1.DF100064  *** OFFLOAD DUMP TO TAPE ***
  SELECT DUMP OFFLOAD
    VOLID 111111 SYS009
    ERASE NO     <<<Specify "ERASE YES" if dump is no longer required on-line >>>
    RETURN
  DUMP NAME SYSDUMP.F1.DF100065  *** OFFLOAD DUMP TO TAPE ***
  SELECT DUMP OFFLOAD
    VOLID 111111 SYS009
    ERASE NO
    RETURN
  SELECT END
/*
/&
```

Process Stand-alone Dump Tape using INFOANA

Delete dumps no longer required on-line.

```
// JOB INFOANAL DELETE S/A DUMP FROM SYSDUMP LIBRARY
/* NEW DUMP LABELS
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
// ASSGN SYS010,DISK,TEMP,VOL=vvvvvv,SHR
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0
// EXTENT SYS010,,1,0,nn,mmm
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',99/365,SD
// EXTENT SYS010,,1,0,nn,mmm
/* END OF DUMP LABELS
// EXEC INFOANA,SIZE=300K
  SELECT DUMP MANAGEMENT              *** DELETE DUMP ***
    DUMP NAME SYSDUMP.BG.DBG00006
      DELETE
    DUMP NAME SYSDUMP.BG.DBG00005
      DELETE
    DUMP NAME SYSDUMP.F2.DF200002
      DELETE
    DUMP NAME SYSDUMP.F2.DF200001
      DELETE
  SELECT END
/*
/&
```

RETURN TO INDEX