

Transaction Debugging Using Dumps

Noel Javier C. Sales
Software Engineer zCICS
Development



IBM CICS® User Conference 2009

Contents

- **Analyzing a transaction dump**
- **ASRA abend cookbook**
- **Locating the failure**
- **Finding the last EXEC CICS command**
- **Cookbook to locate storage fields**
- **Locating relevant data**
- **Cookbook to locate information involved in a link**
- **Fault Analyzer overview**
- **Analyzing a transaction dump with Fault Analyzer**
- **Transaction dump table**
- **Printing transaction dumps**

Introduction

- **This presentation demonstrates the methodology for diagnosing a program check (abend ASRA) in a CICS COBOL application.**
- **A similar approach may be taken when dealing with abends in applications written in other languages supported by CICS, e.g. Assembler or PL1.**
- **The presentation shows how to approach the failure using CICS diagnostic data, coupled with COBOL storage contents and program listing information...**
- **The presentation also shows the ways that Fault Analyzer can be used to assist with problem determination**

Why COBOL?

- **Y2K demonstrated that the COBOL programming language is still very pervasive**
- **Very many COBOL programs have been developed over the years, and inherited by different programmers from those who wrote them initially**
- **An application maintenance policy requires problem determination skills and techniques**
- **Straw poll - how many sites have the need for COBOL experience?**

Using CICS Transaction Dumps

- **Use the transaction dump to locate:**
 - ▶ The point of failure
 - ▶ PSW address
 - ▶ The failing module
 - ▶ The failing instruction
 - ▶ The last EXEC CICS command
 - ▶ Relevant data areas
 - ▶ Called program information
 - ▶ Linking program information

Dump analysis

- **Problem determination analysis is a skill like any other**
 - ▶ It requires time, experience, and education in order to be successful
- **Knowing how to approach a CICS dump is a useful skill to possess**
 - ▶ Once you appreciate dump content, format, etc, it is a good grounding on how to approach CICS (and application) problems in the future
- **This presentation includes a manual approach to dump analysis**
- **There are tools and products available to assist in problem determination**
 - ▶ For example, IBM's Fault Analyzer – providing transaction dump and system dump analysis
 - Fault Analyzer gets control through exits at abend time
 - Transaction abends are analyzed in real time and a report produced
 - Abend information can be saved in a “history file” for later analysis
- **The presentation also includes a review of using Fault Analyzer for transaction dumps**

Traditional ways to analyze CICS dumps

- **CICS generates two types of dump:**
 - Transaction dumps (when a specific CICS task abends, or requests a dump)
 - System dumps (SDUMPs), when the CICS region itself is dumped
- **CICS supplies a batch program for transaction dump formatting**
 - DFHDU650 (for CICS TS 3.2)
- **IPCS is provided by z/OS for SDUMP analysis**
 - IPCS has both an ISPF interaction, and a batch interface
 - It analyses the dump and formats out key state data and control blocks
 - It allows products to provide dump formatting exits for their own control blocks
- **CICS provides a VERBEXIT for IPCS SDUMP formatting**
 - e.g. VERBX DFHPD650 'KE'
- **DFHDU650 and IPCS are static formatters (no dynamic navigation)**

ASRA Abend Cookbook

- FROM THE DUMP...
- 1. Determine the type of program check that occurred.
- Read in CICS Problem Determination Guide for possible causes.
- 2. Review the information in the Transaction Environment.
- 3. Review the exception trace entries at abend. Pay particular attention to the full trace, as well as the last EXEC CICS command prior to the failure.
- 4. Note the PSW address and the failing instruction length.
- 5. Determine the offset of the failing instruction:
 - ▶ Use CESE output
 - ▶ Refer to DFHAP0001 message and subtract the stub length
 - ▶ Look at OFFSET on *EXC trace entry and subtract stub length
 - ▶ Calculate the offset: Subtract the entry point address from the PSW, then subtract the failing instruction length from this
- 6. Calculate the offset of the last EXEC CICS command (Register 14).
 - ▶ Subtract the program's entry point address from the Register 14 address to determine the offset into the program of the last statement executed.

ASRA Abend Cookbook (continued)

- **PROGRAM LISTING - Equating an offset to an instruction:**
- **1. Once an offset is obtained, refer to the offset table in the program listing.**
- **2. Using the offset table, locate the first hex location larger than the offset obtained.**
- **3. Back up to the previous verb. This is the actual statement you want to review. NOTE THE LINE NUMBER.**
- **Hint. If you are looking for an offset that is register 14 or an EXEC CICS command, the verb in the offset table should be a CALL.**
- **4. Refer to the compile output portion of the program listing to locate the corresponding line number. Here you should have the COBOL statement that relates to the offset obtained from the dump.**
- **5. If the abend is not obvious, follow the "Cookbook to locate storage fields".**

Locate the Point of Failure – PSW and registers

ES37CICS --- CICS TRANSACTION DUMP --- CODE=ASRA TRAN=PAYR ID=1/0005

SYMPTOMS= AB/UASRA PIDS/565501800 FLDS/DFHABAB RIDS/PAYPGM1

PSW & REGISTERS AT TIME OF INTERRUPT

PSW	078D0000	<u>8CB04E80</u>	<u>00060004</u>	00000000				
REGS 0-7	0C410590	0C40F378	0C411160	0CB04E0C	0CB04B78	00000000	00000000	0C4111B0
REGS 8-15	0C410F18	0C410450	0CB04C50	0CB04D1C	0CB04C3C	0C40F2E0	8CB04E5E	00000000

REGISTERS AT LAST EXEC COMMAND

REGS 0-7	0C410590	0C40F378	0C4111B0	0CB04E0C	0CB04B78	00000000	00000000	0C4111B0
REGS 8-15	0C410F18	0C410450	0CB04C50	0CB04D1C	0C40CDB0	0C40F2E0	8CB04E5E	00000000

Locate the Point of Failure – transaction info

Transaction environment for transaction number(0000035)

<u>transaction_id</u> (PAYR)	<u>orig_transaction_id</u> (PAYR)	
<u>initial_program</u> (PAYMENU)	<u>current_program</u> (PAYPGM1)	
<u>facility_type</u> (TERMINAL)	<u>facility_name</u> (DYPG)	<u>Start_code</u> (TP)
<u>netname</u> (IBMXDYPG)	<u>profile_name</u> (DFHCICST)	
<u>userid</u> (CICSUSER)	<u>cmdsec</u> (NO)	<u>ressec</u> (NO)
<u>spurge</u> (NO)	<u>dtimeout</u> (4769679)	<u>tpurge</u> (NO)
<u>taskdatakey</u> (USER)	<u>taskdataloc</u> (ANY)	
<u>twasize</u> (00000)	<u>twaaddr</u> ()	
<u>remote</u> (NO)	<u>dynamic</u> (NO)	
<u>priority</u> (001)	<u>Tclass</u> (NO)	<u>runaway_limit</u> (0020000)
<u>indoubt_wait</u> (YES)	<u>indoubt_wait_mins</u> (000000)	
<u>indoubt_action</u> (????)	<u>cics_uow_id</u> (C01C2CABB379EEE02)	<u>confdata</u> (NO)
<u>system_transaction</u> (NO)	<u>restart_count</u> (00000)	<u>restart</u> (NO)

Locate the Point of Failure - abbreviated trace

```

00035 QR AP 00E1 EIP ENTRY WRITEQ-TS                0004,0C40F2E0 . 2\,08000A02 .... =000113=
00035 QR TS 0201 TSQR ENTRY WRITE                CALC,0C410F18 , 0000000C,YES,MAIN,EXEC =000114=
00035 QR AP 00E1 EIP EXIT WRITEQ-TS                OK                00F4,00000000 ....,00000A02 .... =000118=
00035 QR AP 1942 APLI *EXC* Program-CheckSTART PROGRAM,PAYPGM1,CEDF,FULLAPI,EXEC,NO,0C3478A0,0C409C50 , 0000007,2*=000119=
00035 QR AP 0781 SRP *EXC* ABEND_ASRA                PAYPGM1,FFFFFFFF,CICS =000127=

```

Reviewing the Trace Entries prior to the Abend...

The (edited) Trace Entries show that the transaction (task number 35) had issued an EXEC CICS WRITEQ command to CICS Temporary Storage.

Control returned from CICS to the application (as shown by the EIP EXIT Trace Entry). The next trace entry shows an Exception Trace event (*EXC*) issued by CICS module DFHAPLI. This shows that a Program Check was detected. CICS passes control to the System Recovery Program (DFHSRP) and this issues a further *EXC* trace showing the Abend ASRA.

Since the EIP EXIT trace call showed that control returned from CICS at this point, we know the abend occurred whilst not in CICS code.

For further information, Full Trace rather than Abbreviated can be used...

Locate Point of Failure - full trace

Reviewing the relevant Trace Entries using Full Trace...

```
AP 00E1 EIP ENTRY WRITEQ-TS                                REQ(0004) FIELD-A(0C40F2E0 . 2\ ) FIELD-B(08000A02 ....)
TASK-00035 KE_NUM-002B TCB-QR /0069C168 RET-8CB04E5E TIME-05:28:48.6413453754 INTERVAL-00.0001408750 =000113=
```

```
AP 00E1 EIP EXIT WRITEQ-TS OK                             REQ(00F4) FIELD-A(00000000 ....) FIELD-B(00000A02 ....)
TASK-00035 KE_NUM-002B TCB-QR /0069C168 RET-8CB04E5E TIME-05:28:48.6414570004 INTERVAL-00.0000125000 =000118=
```

```
AP 1942 APLI *EXC* - Program-Check FUNCTION(START_PROGRAM) PROGRAM(PAYPGM1) CEDF_STATUS(CEDF) EXECUTION_SET(FULLAPI)
ENVIRONMENT_TYPE(EXEC) SYNCONRETURN(NO) LANGUAGE_BLOCK(0C3478A0) COMMAREA(0C409C50 , 00000007) LINK_LEVEL(2)
SYSEIB_REQUEST(NO)
TASK-00035 KE_NUM-002B TCB-QR /0069C168 RET-8BF7FE5C TIME-05:28:48.6692091254 INTERVAL-00.0277521250* =000119=
1-0000 00580000 000000DA 00000000 00000000 B81B5D40 00000000 02000100 D7C1E8D7 *.....) .....PAYP*
0020 C7D4F140 0C3D9EA0 0C3D4A50 063D9E01 00000001 01B00202 0CB01E30 0C3478A0 *GM1 .....$&.....*
0040 00002CE8 00000000 0C409C50 00000007 00020002 02020000 *...Y.... &..... *
2-0000 F0C3F461 C1D2C5C1 018400C4 0000FFFF C4C6C8C1 D7D3C9F1 0C003F10 0BDA7080 *OC4/AKEA.d.D...DFHAPLI1.....*
0020 0BE77680 0C336400 00000004 00000004 FF850004 00000000 078D0000 8CB04E80 *.X.....e.....+. *
0040 00060004 00000000 8CB04E80 80000000 0C410590 0C40F378 0C411160 0CB04E0C *.....+. .... 3....-..+. *
0060 0CB04B78 00000000 00000000 0C4111B0 0C410F18 0C410450 0CB04C50 0CB04D1C *.....&.<&..(. *
0080 0CB04C3C 0C40F2E0 8CB04E5E 00000000 006FB01F 00000002 00000000 00000000 *..<.. 2\..+;...?.....*
```

```
AP 0781 SRP *EXC* - ABEND_ASRA PROGRAM(PAYPGM1 ) OFFSET(FFFFFFFF) EXEC_KEY(CICS)
TASK-00035 KE_NUM-002B TCB-QR /0069C168 RET-8BE60322 TIME-05:28:48.6694091254 INTERVAL-00.0000113750 =000127=
1-0000 D7C1E8D7 C7D4F140 F0C3F461 C1D2C5C1 FFFFFFFF 00030781 *PAYPGM1 OC4/AKEA.....a *
2-0000 00 * *
3-0000 F0C3F461 C1D2C5C1 018400C4 0000FFFF C4C6C8C1 D7D3C9F1 0C003F10 0BDA7080 *OC4/AKEA.d.D...DFHAPLI1.....*
0020 0BE77680 0C336400 00000004 00000004 FF850004 00000000 078D0000 8CB04E80 *.X.....e.....+. *
0040 00060004 00000000 8CB04E80 80000000 0C410590 0C40F378 0C411160 0CB04E0C *.....+. .... 3....-..+. *
0060 0CB04B78 00000000 00000000 0C4111B0 0C410F18 0C410450 0CB04C50 0CB04D1C *.....&.<&..(. *
0080 0CB04C3C 0C40F2E0 8CB04E5E 00000000 006FB01F 00000002 00000000 00000000 *..<.. 2\..+;...?.....*
```

Locate the Point of Failure - the PSW

Relating a PSW to a Load Module

PSW & REGISTERS AT TIME OF INTERRUPT

```

PSW                078D0000      8CB04E80      00060004      0BB45C00
                    |              |      |
                    |              |      |
                    |              |      |-----> Interrupt code
                    |              |
                    |              |-----> Instruction Length
                    |
                    Address of next instruction
                    to be executed
    
```

PROGRAM INFORMATION FOR THE CURRENT TRANSACTION

Number of Levels 00000002

INFORMATION FOR PROGRAM AT LEVEL 00000002 of 00000002

```

Program Name      PAYPGM1      Invoking Program PAYMENU
Load Point        0CB01E30      Program Length   00002CE8
Entry Point       8CB01E50      Addressing Mode  AMODE 31
Language Defined  Unknown      Language Deduced Unknown
Commarea Address  0C409C50      Commarea Length  00000007
Execution Key     USER        Data Location    ANY
Environment       User application
    
```

Adding the program length to the Load Point address gives an address of 0CB04B18. Yet - this is lower than the PSW address. This means the abend must have occurred outside the current program, as known to CICS.

Locate the Point of Failure - the load module

Relating a PSW to a Load Module (continued)

LOAD LIST ELEMENT

Program Name	BONUSCK	Entry Point	8CB04B40		
00000000	0C369070	0BE99088	0C3D4978	8CB04B40	*.....Z.h..... * 0C369160

----- MODULE INDEX -----

LOAD PT.	NAME	ENTRY PT	LENGTH	LOAD PT.	NAME	ENTRY PT	LENGTH
0C2CC790	DFHCRR	0C2CC7B0	000012B0	0C7E60A0	DFHEITMT	0C7E60A0	00009950
0C2D1000	DFHZATS	0C2D1028	00003218	0C9D8000	DFHAMP	0C9D8020	000243A0
0C2D4220	DFHZXST	0C2D4240	00002438	0CA00000	CEEEV007	0CA00000	00065828
0C2D6660	DFHZNAC	0C2D6680	0000A438	0CA66000	DFHEMTD	0CA66028	00018450
0C2E0AA0	DFHZXRE	0C2E0AC0	00000E08	0CB00000	PAYMENU	0CB00020	00001E28
0C2E18B0	DFHZATA	0C2E18D8	000048A0	0CB01E30	PAYPGM1	0CB01E50	00002CE8
0C2E6150	DFHQRY	0C2E6190	00000F00	0CB04B20	<u>BONUSCK</u>	<u>0CB04B40</u>	<u>000019B8</u>

Using the Load List Elements, a program can be found that has an entry point just prior to the PSW address.

Using this information with the Module Index from the dump, it can be seen that program BONUSCK has a Load Point and Program Length that cover the address where the PSW is pointing. This shows that BONUSCK was in control when the abend ASRA occurred...

PSW address CB04E80 - CB04B40 (BONUSCK entry point address) = 340.

340 - 6 (instruction length) = 33A. Therefore, the abend occurred at X'33A' in program BONUSCK.

Locate Point of Failure - msgs and *EXC* trace

Quick methods to locate the Failing Offset

FROM THE TRACE TABLE

```

AP 0781 SRP *EXC* - ABEND_ASRA PROGRAM(PAYPGM1 ) OFFSET(FFFFFFFF) EXEC_KEY(CICS)
TASK-00035 KE_NUM-002B TCB-QR /0069C168 RET-8BE60322 TIME-05:28:48.6694091254 INTERVAL-00.0000113750 =000127=
1-0000 D7C1E8D7 C7D4F140 F0C3F461 C1D2C5C1 FFFFFFFF 00030781 *PAYPGM1 0C4/AKEA.....a *
2-0000 00 *. *
3-0000 F0C3F461 C1D2C5C1 018400C4 0000FFFF C4C6C8C1 D7D3C9F1 0C003F10 0BDA7080 *0C4/AKEA.d.D...DFHAPLI1.....*
0020 0BE77680 0C336400 00000004 00000004 FF850004 00000000 078D0000 8CB04E80 *.X.....e.....+.*
0040 00060004 00000000 8CB04E80 80000000 0C410590 0C40F378 0C411160 0CB04E0C *.....+.....3.....-...+.*
0060 0CB04B78 00000000 00000000 0C4111B0 0C410F18 0C410450 0CB04C50 0CB04D1C *.....&.<&..(*
0080 0CB04C3C 0C40F2E0 8CB04E5E 00000000 006FB01F 00000002 00000000 00000000 *..<.. 2\..+;.....?.....*
00C0 00000000 00000000 806FA03C 00000001 FF850004 00000000 078D0000 8CB04E80 *.....?.....e.....+.*
00E0 00060004 00000000 8CB04E80 80000000 0C410590 0C40F378 0C411160 0CB04E0C *.....+.....3.....-...+.*
0100 0CB04B78 00000000 00000000 0C4111B0 0C410F18 0C410450 0CB04C50 0CB04D1C *.....&.<&..(*
0120 0CB04C3C 0C40F2E0 8CB04E5E 00000000 006FB01F 00000002 00000000 00000000 *..<.. 2\..+;.....?.....*
    
```

CICS SYSTEM LOG

+DFHAP0001 ES37CICS An abend (code 0C4/AKEA) has occurred at offset X'FFFFFFFF' in module PAYPGM1.

CICS TD QUEUE- CEEMSG.

CEE3204S The system detected a protection exception.

From compile unit BONUSCK at entry point BONUSCK at compile unit offset +0000033A at address 0CB04E7A.

To obtain the correct offset if using the DFHAP0001 message or trace the program stub size and the length of the bad instruction must be subtracted from the offset. The COBOL stub size is the difference between the entry and load points. In this particular case, the offset is shown as x'FFFFFFFF', indicating CICS was not able to determine the failing program, as a dynamic call had taken place.

The Last EXEC CICS Command (R14)

PSW and registers...

PSW & REGISTERS AT TIME OF INTERRUPT

REGS 0-7	0C410590	0C40F378	0C411160	0CB04E0C	0CB04B78	00000000	00000000	0C4111B0
REGS 8-15	0C410F18	0C410450	0CB04C50	0CB04D1C	0CB04C3C	0C40F2E0	8CB04E5E	00000000

REGISTERS AT LAST EXEC COMMAND

REGS 0-7	0C410590	0C40F378	0C4111B0	0CB04E0C	0CB04B78	00000000	00000000	0C4111B0
REGS 8-15	0C410F18	0C410450	0CB04C50	0CB04D1C	0C40CDB0	0C40F2E0	<u>8CB04E5E</u>	00000000

Trace Entries for Dumping Transaction...

```
AP 00E1 EIP EXIT WRITEQ-TS OK                                REQ(00F4) FIELD-A(00000000 ....) FIELD-B(00000A02 ....)
TASK-00035 KE_NUM-002B TCB-QR /0069C168 RET-8CB04E5E TIME-05:28:48.6414570004 INTERVAL-00.0000125000 =000118=
```

```
AP 1942 APLI *EXC* - Program-Check FUNCTION(START_PROGRAM) PROGRAM(PAYPGM1) CEDF_STATUS(CEDF) EXECUTION_SET(FULLAPI)
ENVIRONMENT_TYPE(EXEC) SYNCONRETURN(NO) LANGUAGE_BLOCK(0C3478A0) COMMAREA(0C409C50 , 00000007)
LINK_LEVEL(2) SYSEIB_REQUEST(NO)
```

The Last EXEC CICS Command (R14) continued

Control blocks for the dumping transaction...

TASK CONTROL AREA (SYSTEM AREA) --- TCAPCHS is at offset X'40'

```
000000 00000000 00000000 00000000 00000000 0000035C 0BDC9EE4 00000042 00000000 *.....*...U.....* BE77780
000020 00000000 00000000 00000000 00000000 00000000 00000000 0C33E720 00000000 *.....X.....* BE777A0
000040 0C40F2E0 00000000 00000000 00000000 00000000 00000000 00000000 *. 2.....* BE777C0
```

TRANSACTION STORAGE-USER31 ADDRESS 0C40AB70 TO 0C411D5F LENGTH 000071F0

```
004760 00000000 00000000 00000000 00000000 00104001 0C40EFD8 00000000 8CB04E5E *..... .Q.....+;* C40F2D0
004780 00000000 0C410590 0C40F378 0C4111B0 0CB04E0C 0CB04B78 00000000 00000000 *..... 3.....+.....* C40F2F0
0047A0 0C4111B0 0C410F18 0C410450 0CB04C50 0CB04D1C 0C40CDB0 00000000 0C40F3A8 *.....<... (... .. 3y* C40F310
0047C0 00000000 00000000 0C40F2E0 0C410450 00000000 00000000 00000000 *..... 2.....* C40F330
```

----- MODULE INDEX -----

LOAD PT.	NAME	ENTRY PT	LENGTH	LOAD PT.	NAME	ENTRY PT	LENGTH
0C2D6660	DFHZNAC	0C2D6680	0000A438	0CA66000	DFHEMTD	0CA66028	00018450
0C2E0AA0	DFHZXRE	0C2E0AC0	00000E08	0CB00000	PAYMENU	0CB00020	00001E28
0C2E18B0	DFHZATA	0C2E18D8	000048A0	0CB01E30	PAYPGM1	0CB01E50	00002CE8
0C2E6150	DFHQRY	0C2E6190	00000F00	0CB04B20	BONUSCK	0CB04B40	000019B8

Register 14 in any case is equal to 0CB04E5E. This address also falls into program BONUSCK whose entry point is 0CB04B40. Subtracting the entry point from Register 14, we get an offset of 31E.

Locate Instructions using the program offset

The offset table in the program listing

LINE #	HEXLOC	VERB	LINE #	HEXLOC	VERB	LINE #	HEXLOC	VERB
000067	0002D6	MOVE	000068	0002DC	CALL	000075	00032A	DIVIDE
000079	000340	IF	000080	00035E	MOVE	000083	000364	GOBACK

Relate the offset to the actual compile line

```

000019      LINKAGE SECTION.
000055      01  PAYPASS.                                BLL=0003+000
000056          10 BONI-CHECK                        PIC 9(7).        BLL=0003+000
000057          10 SALARY-CK                          PIC S9999999V99 USAGE COMP-3.  BLL=0003+007
000058      01  PAYCALC.                                BLL=0004+000
000059          10 MONTHLY-SALARY                    PIC S999999V99  USAGE COMP-3.  BLL=0004+000
000060      *
000061      PROCEDURE DIVISION USING DFHEIBLK DFHCOMMAREA PAYPASS.
000062      *
000063      CALCULATE-BONUS.
000064      *EXEC CICS WRITEQ TS QUEUE('CALC')
000065      *                                FROM(PAYPASS) MAIN
000066      *END-EXEC.
000067      Move length of PAYPASS to dfhb0020
000068      Call 'DFHEI1' using by content x'0a02e0000700004900f0f0f0f2f5
000069      - '404040' by content 'CALC' by reference PAYPASS by
000070      reference dfhb0020 end-call.
000071
000072      *
000073      * CALCULATE MONTHLY SALARY
000074      *
000075      DIVIDE SALARY-CK BY 12 GIVING MONTHLY-SALARY.
000076      *
000077      * IF BONUS AMOUNT IS LARGER THAN MONTHLY SALARY THEN ERROR
    
```

Locate Instructions Using Offset (continued)

- The PSW address points to the instruction at offset X'33A'. Looking at the 'HEXLOC' column, 340 is the first offset larger than 33A. Backing up to the previous verb, we see that #75 is the line number where the failure occurred (PSW calculated offset).
- The Register 14 calculated offset is 31E. Following the same procedure above, we see that the VERB is a CALL and the COBOL line number is 68. Backing up to the first outdented statement, we see that an EXEC CICS WRITEQ TS command was issued.

ASRA Abend Cookbook worksheet

- FROM THE DUMP...
- 1. Determine the type of program check that occurred.
 - ▶ An 0C4 occurred. This is a protection exception problem.
- 2. Review the information in the Transaction Environment.
 - ▶ Transaction environment shows the current program, userid, how invoked, etc.
- 3. Review the exception trace entries.
 - ▶ The exception trace entry showed that an ASRA 0C4 occurred.
 - ▶ The offset calculated was not a good one, it was FFFFFFFF.
 - ▶ The last EXEC CICS command was an EXEC CICS WRITEQ TS.
 - ▶ CICS suspected that the program in control at the time of the failure was PAYPGM1.
- 4. Note the PSW address and failing instruction length.
 - ▶ The PSW address was CB04E80 and the failing instruction length was 6.
- 5. Locate the offset of the failing instruction.
 - ▶ Subtract the entry point address from the PSW address.
 - ▶ Entry point of CB04B40 subtracted from PSW address CB04E80 gives an offset of 340.
 - ▶ Subtract the assembler instruction length from the answer above.
 - ▶ 6 (on PSW line - interrupt information) subtracted from 340 gives 33A.
- 6. Calculate the offset of the last EXEC CICS command (Register 14).
 - ▶ Subtract the entry point address from the R14 address
 - ▶ Entry point of CB04B40 subtracted from R14 CB04E5E gives an offset of 31E.

ASRA Abend Cookbook worksheet (continued)

- FROM THE PROGRAM LISTING...
- 1. Using the offset table, locate the first hexloc larger than the offset obtained for the PSW and register 14.
- 2. PSW offset of 33A.
 - ▶ The first hexloc larger than 33A is 340.
- 3. Back up to the previous COBOL verb. This is the failing statement. Note the LINE #.
 - ▶ Backing up to the previous verb, we see the line number is 75.
- 4. Refer to the compile listing of the program to locate the corresponding line number.
 - ▶ The statement where the abend occurred is 'DIVIDE SALARY-CK BY 12 GIVING MONTHLY-SALARY'.
- 5. Register 14 offset is 31E.
 - ▶ The first hexloc larger than 31E is 32A.
- 6. Back up to the previous COBOL verb. This is the last EXEC CICS statement. Note the LINE # of 68.
- 7. Refer to the compile listing of the program to locate the corresponding line number. For a CICS command, look back up at the first "out-dented" line to see the actual command.
 - ▶ The last EXEC CICS statement was EXEC CICS WRITEQ TS QUEUE('CALC') FROM(PAYPASS) MAIN END-EXEC.
- 8. If the abend is not obvious, follow the "Cookbook to locate storage fields".

Locating Relevant Data Areas

- **COBOL has two types of data areas:**
 - ▶ Working storage
 - Data items are in contiguous storage
 - Storage is pre-allocated
 - Addressed by Base Locator cells (BLW or BL)
- **Linkage section**
 - ▶ Storage passed to a program
 - ▶ Storage acquired by a program
 - ▶ Addressed by Base Locator (BLL)
- **Each 01 level is addressed by a different BLL cell**
- **Base locator cells reside in the COBOL Task Global Table (TGT).**

Checklist to find TGT address (COBOL for z/OS)

- Approach 1
 - ▶ Use Register 9 address from 'REGISTERS AT TIME OF INTERRUPT' or from 'REGISTERS AT LAST EXEC COMMAND'
- Approach 2
 - ▶ Record Register 13 address from 'REGISTERS AT TIME OF INTERRUPT' or from 'REGISTERS AT LAST EXEC COMMAND'
 - ▶ Add 5C to this address. Go there in the dump. At that storage location will be an address that is the TGT address.
- Approach 3
 - ▶ Go to the trace table. Look for an EXEC CICS command that your program issued (EIP ENTRY). Write down the Field A address.
 - ▶ Add 5C to this address. Go there in the dump. At that storage location will be an address that is the TGT address.
- Approach 4
 - ▶ Obtain the address in field TCAPCHS (from the System TCA)
 - ▶ Add 5C to this address. Go there in the dump. At that storage location will be an address that is the TGT address.
- Approach 5
 - ▶ If the LE run-time option TRMTHDACT=UADUMP is set, look at the CEEMSG output. The TGT address will be identified.

Finding the TGT Address in a Dump...

Approach 1 and 2:

PSW & REGISTERS AT TIME OF INTERRUPT

```
PSW          078D0000   8CB04E80   00060004   00000000
REGS 0-7     0C410590   0C40F378   0C411160   0CB04E0C           0CB04B78   00000000   00000000   0C4111B0
REGS 8-15    0C410F18   0C410450   0CB04C50   0CB04D1C           0CB04C3C   0C40F2E0   8CB04E5E   00000000
```

Approach 3:

== TRACE ENTRIES FOR DUMPING TRANSACTION ==

```
00035 QR  AP 00E1 EIP  ENTRY WRITEQ-TS                0004,0C40F2E0. 2\,08000A02 .... =000113=

AP 00E1 EIP ENTRY WRITEQ-TS                REQ(0004) FIELD-A(0C40F2E0 . 2\) FIELD-B(08000A02 ....) TASK-00035 KE_NUM-002B TCB-QR
/0069C168

RET-8CB04E5E TIME-05:28:48.6413453754 INTERVAL-00.0001408750 =000113=
```

Approach 4:

TASK CONTROL AREA (SYSTEM AREA) TCAPCHS is at offset X'40' (pointer to high level save area)

```
000000 00000000 00000000 00000000 00000000 0000035C 0BDC9EE4 00000042 00000000 *.....*...U.....* 0BE77780
000020 00000000 00000000 00000000 00000000 00000000 00000000 0C33E720 00000000 *.....X.....* 0BE777A0
000040 0C40F2E0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *. 2.....* 0BE777C0
```

Approach 2 and 4:

TRANSACTION STORAGE-USER31 ADDRESS 0C40AB70 TO 0C411D5F LENGTH 000071F0

```
000000 E4F0F0F0 F0F0F3F5 00000000 00000000 00000000 8C00ABA8 00026B98 0BC4B418 *U0000035.....y.,q.D.* 0C40AB70
004720 C2D6D5E4 E2C3D240 81000000 0C40F288 0C40F89C 94000000 0C40F288 00000000 *BONUSCK a.... 2h. 8.m.... 2h....* 0C40F290
004780 00000000 0C410590 0C40F378 0C4111B0 0CB04E0C 0CB04B78 00000000 00000000 *..... 3.....+.....* 0C40F2F0
0047A0 0C4111B0 0C410F18 0C410450 0CB04C50 0CB04D1C 0C40CDB0 00000000 0C40F3A8 *.....<... (.. ..... 3y* 0C40F310
0047C0 00000000 00000000 0C40F2E0 0C410450 00000000 00000000 00000000 00000000 *..... 2..... 30* 0C40F330
```

Finding the TGT Address (continued)

Approach 5:

CESEMSG OUTPUT (TRMTHDACT=UADUMP)

Information for enclave PAYPGM1 called by enclave PAYMENU

Program BONUSCK was compiled 02/05/03 5:00:31 AM

TGT for BONUSCK: 0C410450

Program PAYPGM1 was compiled 02/05/03 2:52:01 AM

TGT for PAYPGM1: 0C410BC0

Storage and control blocks for non-active routines:

TGT for PAYMENU : 0C4099F0

Cookbook to Locate Storage Fields

- **From the compile listing for the program...**
- **1. Go to the Data Division Map and note the following:**
 - ▶ The Field you wish to locate
 - ▶ The BLW/BLL number
 - ▶ The hex displacement of the field
 - ▶ Data type and the number of bytes
- **2. Find the TGT Memory Map at the end of the compile listing. Record the offset within the TGT where the Base Locators for Working Storage (BLW) or Base Locators for Linkage (BLL) can be found.**

- **From the dump...**
- **1. Obtain the TGT address.**
- **2. Add the offset from step 2 above to the TGT address. Go there in the dump. This will point to another address which is the first base locator for working storage (BLW0) or the first base locator for linkage (BLL0).**
- **3. If the field you wish to find is located off of BLW0/BLL0 then note this address. If it is located off of BLW1/BLL1 then note the address in the next word, BLW2/BLL2 the third word... etc.**
- **4. Add the displacement of the field, from the compile listing, to the base locator address found in the previous step.**
- **5. Go to the address obtained in the previous step. Starting at this address, highlight the number of bytes for the field as defined in the program listing.**

The Program Listing

The Data Division map

Source LineID	Hierarchy and Data Name	Base Locator	Hex-Displacement Blk	Structure	Asmblr Data Definition	Data Type
3	PROGRAM-ID BONUSCK-----*					
55	1 PAYPASSBLL=0003	000		DS 0CL12	Group
56	2 BONI-CHECK.BLL=0003	000	0 000 000	DS 7C	Disp-Num
57	2 SALARY-CKBLL=0003	007	0 000 007	DS 5P	Packed-Dec
58	1 PAYCALCBLL=0004	000		DS 0CL5	Group
59	2 MONTHLY-SALARY.BLL=0004	000	0 000 000	DS 5P	Packed-Dec

TGT Memory Map

TGTLOC.		*** VARIABLE PORTION OF TGT ***
		000120 BASE LOCATORS FOR SPECIAL REGISTERS
000000 RESERVED - 72 BYTES		<u>000128 BASE LOCATORS FOR WORKING STORAGE</u>
000048 TGT IDENTIFIER		<u>00012C BASE LOCATORS FOR LINKAGE SECTION</u>
00004C RESERVED - 4 BYTES		000140 CLLE ADDR. CELLS FOR CALL LIT. SUB-PGMS.
000050 TGT LEVEL INDICATOR		000144 INTERNAL PROGRAM CONTROL BLOCKS
000051 RESERVED - 3 SINGLE BYTE FIELDS		
000054 32 BIT SWITCH		*** DSA MEMORY MAP ***
000058 POINTER TO RUNCOM		DSALOC
00005C POINTER TO COBVEC		
000064 NUMBER OF FCB'S		000000 REGISTER SAVE AREA
000068 WORKING-STORAGE LENGTH		00004C STACK NAB (NEXT AVAILABLE BYTE)
000114 WORKING-STORAGE ADDRESS		000058 ADDRESS OF INLINE-CODE PRIMARY DSA
000118 POINTER TO FIRST SECONDARY FCB CELL	00005C	ADDRESS OF TGT
00011C POINTER TO STATIC CLASS INFO BLOCK		000080 PROCEDURE DIVISION RETURNING VALUE

Transaction Dump Areas - Registers and Transaction Storage...

PSW & REGISTERS AT TIME OF INTERRUPT

```

PSW          078D0000  8CB04E80  00060004  00000000
REGS 0-7     0C410590  0C40F378  0C411160  0CB04E0C          0CB04B78  00000000  00000000  0C4111B0
REGS 8-15    0C410F18  0C410450  0CB04C50  0CB04D1C          0CB04C3C  0C40F2E0  8CB04E5E  00000000
    
```

EXEC INTERFACE BLOCK.

```

000000  0052848F 0099036F D7C1E8D9 0000035C  C4E8D7C7 00000660 00077D0A 02000000  *..d..r.?PAYR...*DYPG...-'.....* 0C4000D0
000020  000000D7 C1E8D9D6 D3D34000 00000000  000000C3 C1D3C340 40404000 00000000  *...PAYROLL .....CALC .....* 0C4000F0
    
```

PROGRAM COMMUNICATION AREA

```

                                ADDRESS 0C409C50 TO 0C409C56  LENGTH 00000007
000000  40F0F0F0 F0F1F0                                * 000010                                * 0C409C50
    
```

TRANSACTION STORAGE-USER31

```

                                ADDRESS 0C40AB70 TO 0C411D5F  LENGTH 000071F0

000000  E4F0F0F0 F0F0F3F5 00000000 00000000  00000000 8C00ABA8 00026B98 0BC4B418  *U0000035.....y..,q.D..* 0C40AB70

0059C0  00000000 00000000 0CB04C3C 00000001  0C410594 0C410408 0CB04C9C 00000000  *.....<.....m.....<.....* 0C410530
0059E0  0CB04B40 0CB04C58 0C410594 0CB04C4C  00000000 0C4111B0 00000000 00000000  *... ..<...m..<<.....* 0C410550
005A00  00000000 0C411160 0C4111B0 00000000  0C4000D0 0C409C50 0C410F18 00000000  *.....-..... .. ..* 0C410570
005A20  00000000 40000000 00000000 00000000  00000000 00000001 0C410450 0C411160  *..... .. ..* 0C410590
006080  00000000 00000000 00000000 00000000  00000000 00000000 F3E3C7E3 00000000  *.....3TGT.....* 0C410BF0
0060A0  05000000 60030220 0C410818 000D4C2C  0C410D40 00000000 00000362 00000000  *.....<..... .. ..* 0C410C10
0060C0  00000000 0C410D78 00000000 00000000  0C40CDB0 0000017C 00000000 00000000  *..... .. ..* 0C410C30
0060E0  00000000 00000001 E2E8E2D6 E4E34040  C9C7E9E2 D9E3C3C4 00000000 00000000  *.....SYSOUT IGZSRCTD.....* 0C410C50
006100  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *..... .. ..* 0C410C70
    
```

Worksheet to Locate Storage Fields

- From the compile listing for the program:
 - 1. Go to the Data Division Map and note the following:
 - ▶ SALARY-CK uses BLL 3 plus a displacement of 7. It is a packed numeric data type 5 characters long.
 - ▶ MONTHLY-SALARY uses BLL 4 plus a displacement of 0. It is a packed numeric data type 5 characters long.
 - 2. Find the TGT Memory Map at the end of the compile listing and record the offset within the TGT where the Base Locators for Linkage (BLL) can be found.
 - ▶ The first base locator for linkage can be found at offset 12C from the beginning of the TGT.

Worksheet to Locate Storage Fields (continued)

- **From the dump:**
- **1. Obtain the TGT address**
 - ▶ The address of the TGT from R9 on page 1 of the dump is C410450.
- **2. Add the offset of the BLL cells to the TGT address obtained above, and go there. This will point to another address, which is the first BLL.**
 - ▶ Adding 12C to C410450 gives C41057C. Going to that storage location we find an address of 00000000. This is BLL 0.
- **3. If the field you wish to find is located off of BLL 0 then note this address.**
 - ▶ We want BLL 3 and BLL 4, so go over 4 words for BLL 3 and you see that the address is C410F18. The next word is BLL 4, but it is 00000000. THIS IS THE CAUSE OF THE 0C4!
- **4. Add the displacement of the field, from the compile listing, to the base locator found in the previous step.**
 - ▶ Adding the displacement of SALARY-CK (7) to BLL 3 (C410F18), we get C410F1F.
- **5. Go to the address obtained in the previous step. Starting at this address, highlight the number of bytes for the field - as defined in the program listing.**
 - ▶ Looking at storage at address C410FF for 5 bytes, we find that the data is 005275000C. This data looks fine.

A final look at the program...

PAYCALC was not passed as an external reference when BONUSCK was invoked, nor was storage acquired for it at runtime via a GETMAIN or a READ SET command. As such, nothing mapped the 01 structure to its actual storage, and BLL 4 remained 0.

```

000019 LINKAGE SECTION.
000055 01 PAYPASS.                                BLL=0003+000
000056 10 BONI-CHECK          PIC 9(7).          BLL=0003+000
000057 10 SALARY-CK           PIC S9999999V99  USAGE COMP-3.  BLL=0003+007
000058 01 PAYCALC.                                BLL=0004+000
000059 10 MONTHLY-SALARY      PIC S999999V99  USAGE COMP-3.  BLL=0004+000
000060 *
000061 PROCEDURE DIVISION USING DFHEIBLK DFHCOMMAREA PAYPASS.
000062 *
000063 CALCULATE-BONUS.
000064 *EXEC CICS WRITEQ TS QUEUE('CALC')
000065 *                               FROM(PAYPASS) MAIN
000066 *END-EXEC.
000067     Move length of PAYPASS to dfhb0020
000068     Call 'DFHEI1' using by content x'0a02e0000700004900f0f0f0f2f5
000069 -   '404040' by content 'CALC   ' by reference PAYPASS by
000070     reference dfhb0020 end-call.
000071
000072 *
000073 * CALCULATE MONTHLY SALARY
000074 *
000075     DIVIDE SALARY-CK BY 12 GIVING MONTHLY-SALARY.
000076 *
000077 * IF BONUS AMOUNT IS LARGER THAN MONTHLY SALARY THEN ERROR

```


A final look at the storage in the dump...

BLL cells 0, 1, 2, 3 and 4

```

0059C0 00000000 00000000 0CB04C3C 00000001 0C410594 0C410408 0CB04C9C 00000000 *.....<.....m.....<.....* 0C410530
0059E0 0CB04B40 0CB04C58 0C410594 0CB04C4C 00000000 0C4111B0 00000000 00000000 *...<...m..<<.....* 0C410550
005A00 00000000 0C411160 0C4111B0 00000000 0C4000D0 0C409C50 0C410F18 00000000 *.....-.....* 0C410570
005A20 00000000 40000000 00000000 00000000 00000000 00000001 0C410450 0C411160 *.....* 0C410590
006080 00000000 00000000 00000000 00000000 00000000 00000000 F3E3C7E3 00000000 *.....3TGT.....* 0C410BF0
0060A0 05000000 60030220 0C410818 000D4C2C 0C410D40 00000000 00000362 00000000 *.....<.....* 0C410C10
0060C0 00000000 0C410D78 00000000 00000000 0C40CDB0 0000017C 00000000 00000000 *.....* 0C410C30
0060E0 00000000 00000001 E2E8E2D6 E4E33404 C9C7E9E2 D9E3C3C4 00000000 00000000 *.....SYSOUT IGZSR TCD.....* 0C410C50
006100 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....* 0C410C70
006120 00000000 00000000 00000000 00000000 00000000 00000000 0CB01F4C 00000001 *.....<.....* 0C410C90
006140 0C410D28 0C410AF0 0CB02448 00000000 8CF01E50 0CB01FB0 0C410D28 0CB01F84 *.....0.....d* 0C410CB0
006160 00000000 0C410DD8 00000000 00000000 00000000 0C410D88 0C410DD8 00000000 *.....Q.....h...Q....* 0C410CD0
006180 0C4000D0 0C409C50 00000000 00000000 00000000 00000000 00000000 00000000 *.....* 0C410CF0
0061A0 00000000 00000000 00000000 00000000 00000000 00000000 40000000 00000000 *.....* 0C410D10
0061C0 00000000 00000000 00000001 00000000 0C410BC0 0C410D88 0C410DD8 00000000 *.....h...Q....* 0C410D30
0061E0 C8C1D5C3 0C40BA18 0C40BA18 00000000 8C410D50 0C4111E8 00001000 00000B68 *HANC...Y.....* 0C410D50
006200 0C410D50 000003D8 000003C9 00000000 00000000 00000000 00000000 00000000 *.....Q...I.....* 0C410D70
006220 00000000 00000000 C9C7E9E2 D9E3C3C4 00000000 00000000 00000000 00000000 *.....IGZSR TCD.....* 0C410D90
006240 00000000 00000000 E2E8E2D6 E4E33404 00000000 00000000 0E000000 00000000 *.....SYSOUT.....* 0C410DB0
006260 0F000000 00000000 40404040 40404040 00000000 00000000 00000000 00000000 *.....* 0C410DD0
0062C0 00000000 00000000 000000F0 F0F0F0F1 F0C2D6D5 E4E2C3D2 40000000 00000000 *.....000010BONUSCK.....* 0C410E30
0062E0 F0F0F0F0 F1F00009 C3C8D9C9 E2E3C9D5 C5000000 C90004C8 C1C1E200 00000000 *000010..CHRISTINE...I..HAAS.....* 0C410E50

006300 00000000 0000C1F0 F0F1F2F1 F2F0F161 F0F161F1 F9F6F5D7 D9C5E240 40404000 *.....A00121201/01/1965PRES . * 0C410E70
006320 12C6F0F8 61F1F461 F1F9F3F3 00527500 0C000100 00C00004 22000C00 00000000 *.F08/14/1933.....* 0C410E90
006340 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....* 0C410EB0
0063A0 00000000 00000000 F0F0F0F0 F0F0F500 5275000C 00000000 40404040 40404040 *.....0000005.....* 0C410F10
0063C0 40404040 40404040 40404040 40404040 40404040 404040D7 C8D6D5C5 *.....PHONE* 0C410F30
    
```

Program Linkage information

PROGRAM INFORMATION FOR THE CURRENT TRANSACTION

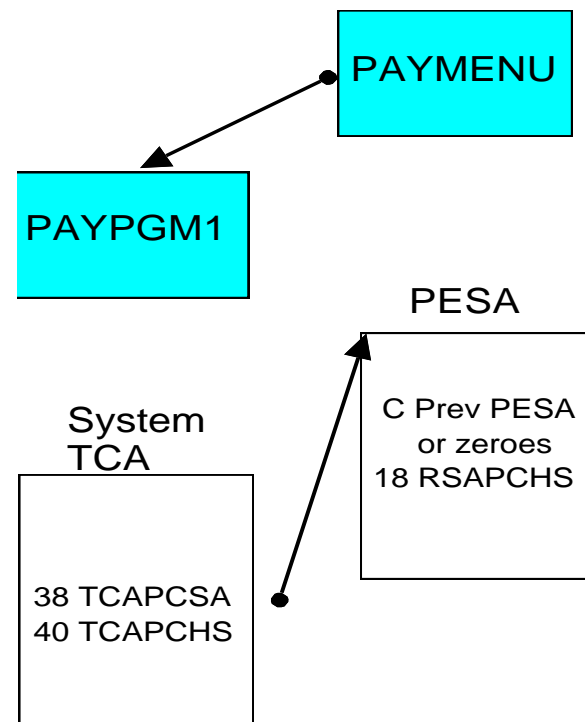
Number of Levels 00000002

INFORMATION FOR PROGRAM AT LEVEL 00000002 of 00000002

Program Name	PAYPGM1	Invoking Program	PAYMENU
Load Point	0CB01E30	Program Length	00002CE8
Entry Point	8CB01E50	Addressing Mode	AMODE 31
Language Defined	Unknown	Language Deduced	Unknown
Commarea Address	0C409C50	Commarea Length	00000007
Execution Key	USER	Data Location	ANY
Environment	User application		

INFORMATION FOR PROGRAM AT LEVEL 00000001 of 00000002

Program Name	PAYMENU	Invoking Program	CICS
Load Point	0CB00000	Program Length	00001E28
Entry Point	8CB00020	Addressing Mode	AMODE 31
Language Defined	Unknown	Language Deduced	Unknown
Commarea Address	0C403988	Commarea Length	00000007
Execution Key	USER	Data Location	ANY
Environment	User application		



Cookbook to Locate Information in a CICS Link

- 1. Locate field TCAPCSA. If it is equal to zeroes, stop. This means that no link is currently active.
- 2. Add x'18' to the address found in TCAPCSA.
- 3. Go to that location in storage. It will be within a heading of KERNEL STACK ENTRY OWNED BY DFHPGLE with an eyecatcher of DFHPESA.
- 4. At that location should be another address. This address can be used just like TCAPCHS to locate the linker's working storage and registers. For COBOL II programs, this IS the TGT address.
- 5. If this is a COBOL for MVS program or higher, add x'5C' to the address found in the previous step and go there in the dump. At that storage location will be another address that is the TGT address.
- 6. Follow the Cookbook to Locate Storage Fields at this point to find any fields you wish.
- 7. If there are more than two links involved, there will be multiple PESA's. To see if there are additional links, take the address in step 1 above and add x'C' to it. If this is zeroes, there are no more links.
- 8. If it is not zero, add x'18' to the address that was found at offset x'C' to find the third program's working storage and registers, and so forth.

Locating Storage involved in a Link

TASK CONTROL AREA (SYSTEM AREA) --- TCAPCSA is at offset X'38'

```
000000 00000000 00000000 00000000 00000000 0000035C 0BDC9EE4 00000042 00000000 *.....*...U.....* 0BE77780
000020 00000000 00000000 00000000 00000000 00000000 00000000 0C33E720 00000000 *.....X.....* 0BE777A0
000040 0C40F2E0 00000000 00000000 00000000 00000000 00000000 00000000 * . 2.....* 0BE777C0
```

KERN STACK ENTRY OWNED BY DFHPGLE

```
000000 00000490 0C33E0C0 00000000 8BF7FE5C 8BB00280 00000410 0C33E4A0 00006000 *.....7.*.....U...-* 0C33E380

...

0003A0 00EC6EC4 C6C8D7C5 E2C10180 00000000 8C2412F4 00000000 0C407C68 00000000 *...>DFHPESA.....4.....* 0C33E720
0003C0 00000000 00070000 00000000 00000C40 39A80000 00000000 00000C34 78280C33 *......y.....* 0C33E740
0003E0 DEE40000 00000000 08000100 00068880 80000000 00000000 00000C40 39F00000 *U.....h......0.* 0C33E760
000400 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*.....* 0C33E780
000420 00000000 00000000 00000C40 00D00C40 39880C40 7C680000 00000000 00000000 *......h.....* 0C33E7A0
000440 00008C00 684E0002 6B980BC4 B4180C33 DE900BE7 79888C00 3F100C33 DB900C00 *.....+,q.D.....X.h.....* 0C33E7C0
000460 4F0F0C00 5F0E0C00 6F0D0C00 7F0C0004 73C80000 00150C34 78280BE7 76800000 *.....?.....H.....X...* 0C33E7E0
000480 00000000 00000000 05330180 0C33EA60 *.....-*.....* 0C33E800
```

TRANSACTION STORAGE-USER31 ADDRESS 0C4039A0 TO 0C40AB6F LENGTH 000071D0

```
000000 E4F0F0F0 F0F0F3F5 8C009E08 0C33E0C0 0C00AE07 00000024 8C347844 0C33E3E4 *U0000035.....TU* 0C4039A0
....
004300 0C4099F0 0CB0012C 0CB002A4 0C405BE0 00000000 0C407D60 00000000 00000000 * . r0.....u. £..... '-.....* 0C407CA0
004320 0C407C68 0C4099F0 00000000 00000000 00000000 00000000 00000000 * . ... r0.....* 0C407CC0
004340 00000000 00000000 00000000 00000000 00000000 00000000 0CB007E4 *.....U* 0C407CE0
```

Worksheet to Locate Information in a Link

- **1. Locate field TCAPCSA within the dump.**
 - ▶ The TCAPCSA address is C33E720. Since this is not zeroes, we know that a link is currently active.
- **2. To locate storage fields for the LINKED FROM program, add x'18' to the address in TCAPCSA.**
 - ▶ Adding x'18' to C33E720 gives us C33E738. Going there in the dump, we find an address of C407C68 (as this is COBOL for MVS, add 5C to this address to then take us to address C407CC4. This contains the TGT address - C4099F0).
 - ▶ This is akin to having TCAPCHS for the LINKED FROM program. To find storage areas for the LINKED FROM program, use the 'Cookbook to Locate the TGT' and the 'Cookbook to Locate Storage Fields' substituting the address we just obtained, C407CC4, for Register 13, the EIP entry or TCAPCHS address (approaches 2 through 4).
- **3. To see whether a third link is involved, take the address in TCAPCSA and add x'C' to it. At that storage location, if it is zeroes, no further links are involved.**
 - ▶ TCAPCSA address of C33E720 plus x'C' gives us C33E72C. This location contains zeroes.

COBOL... COBOL, COBOL,

- Depending on what level of COBOL your program is compiled against, various differences will be seen.

- CICS has supported various flavours of COBOL
 - ▶ OS/VS COBOL
 - ▶ VS COBOL II
 - ▶ IBM COBOL for OS/390 and VM
 - ▶ IBM Enterprise COBOL for z/OS and OS/390

- One noticeable difference is the TGT eyecatcher.
 - ▶ This will be TGT, 2TGT or 3TGT...

- NOTE – 3TGT is the eyecatcher for Enterprise COBOL...

```

00000000 40000000 00000000 00000000 00000000 00000001 0C410450 0C411160 *.....- * 0C410590
00000000 00000000 00000000 00000000 00000000 00000000 F3E3C7E3 00000000 *.....3TGT.... * 0C410BF0
05000000 60030220 0C410818 000D4C2C 0C410D40 00000000 00000362 00000000 *...-.....<..... * 0C410C10
    
```

Additional Reference

http://ew.share.org/client_files/callpapers/attach/SHARE_in_Orlando/S1055RE123730.pdf - checklist for CICS-COBOL debugging

FAULT ANALYZER

Fault Analyzer

- **IBM product to assist with problem determination**

- 5655-U28

- **Provides abend analysis assistance**

COBOL

PL/I

Assembler

C/C++

Language Environment

UNIX System Services

CICS

IMS

DB2

MQSeries

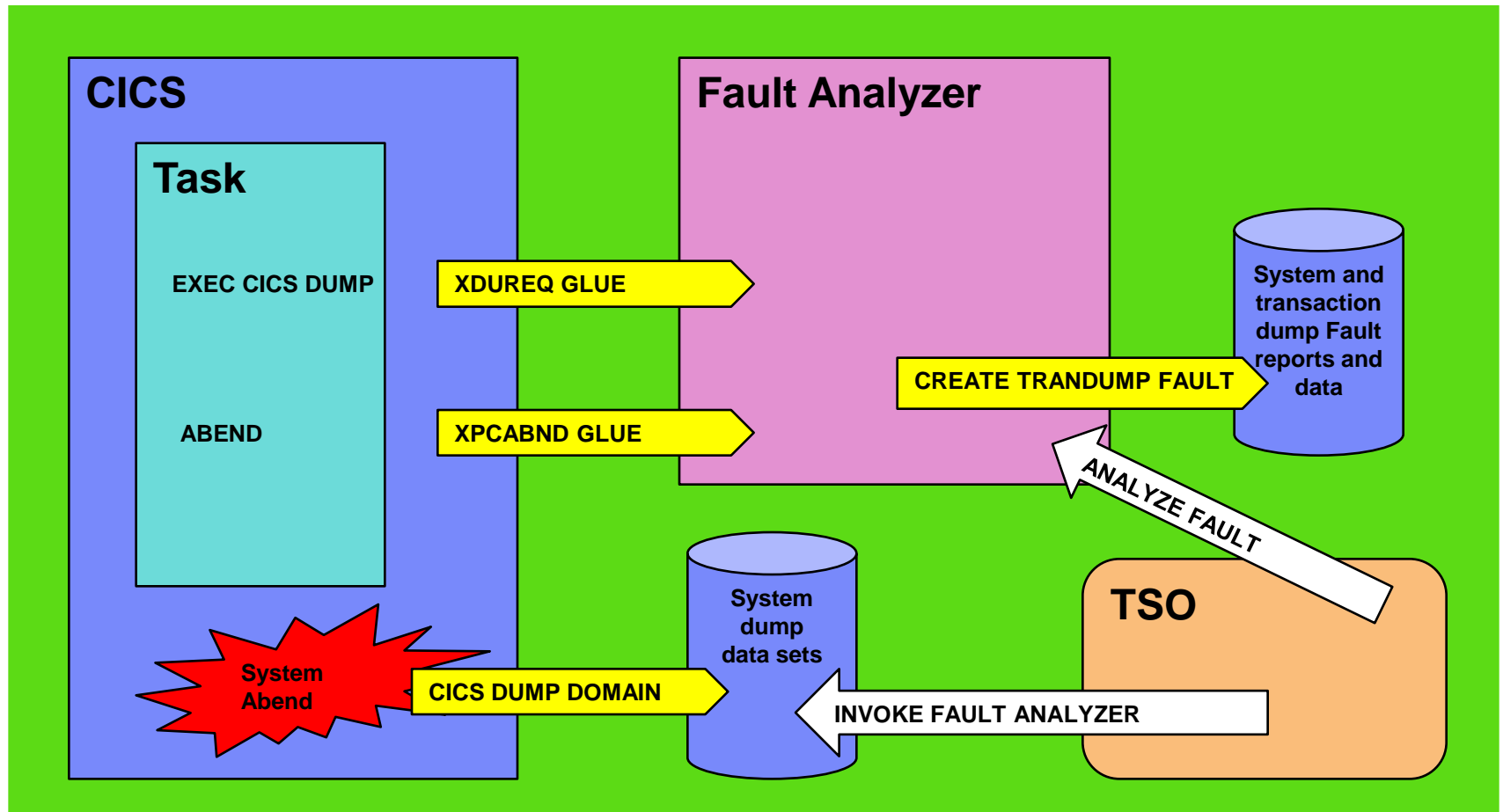
WebSphere

Java

Fault Analyzer and CICS

- **Problem determination and dump analysis assistance for Faults**
- **Fault Analyzer provides diagnostic analysis for CICS**
 - Transaction abends
 - System dumps
- **For transaction abends and dumps, Fault Analyzer is invoked to capture information at the time**
 - Via CICS XPCABND GLUE for transaction abends
 - Via CICS XDUREQ GLUE for EXEC CICS DUMP TRANSACTION commands
 - Via LE abnormal termination exit CEEEXTAN
 - The information and report can be saved in a history file for later analysis
- **For system dumps, Fault Analyzer runs against the SDUMP itself**
 - Fault Analyzer does not need to be installed in the CICS region to do this
 - The dump taken as normal, and analysed / dynamically navigated after the event

How Fault Analyser is used with CICS



Fault Analyzer – setup work

- **Add the Fault Analyzer library to the CICS DFHRPL concatenation**
 - DD DSN=PP.FAULTANL.V810.SIDIAUTH,DISP=SHR
- **Add an entry into PLT for PI and SD:**
 - DFHPLT TYPE=ENTRY,PROGRAM=IDIPLT
 - This allows CICS to invoke Fault Analyzer via its GLUEs at dump or abend time, to capture **transaction** dump information for a Fault report
- **Define the Fault Analyzer programs and map to CICS, or utilise program autoinstall to define them**
 - IDIPLT, IDIPLTD, IDIPLTS, IDIXCX52, IDIXCX53, IDIXFA, IDIXMAP
- **Configure CICS LE abnormal termination CEEEXTAN CSECT exit IDIXCCEE**
 - This exit invokes Fault Analyzer for LE programs

Invoking Fault Analyzer

- You can invoke it from under TSO...
- or explicitly invoke it against a system dump data set (or a history file in ISPF 3.4)

```

OPTION  ==> fau
 1 MVS/DITTO - MVS/DITTO Utility Program
 3 OMVS      - OpenEdition MVS
 4 QCB       - QCBTRACE V58
 5 DB        - DATABASE functions
 6 Terse     - Terse compress/uncompress tool
 8 DFSORT    - Interactive DFSORT
 9 SDF2      - Screen Definition Facility II
APM AppMonitor - Application Monitor for z/OS ( NEW )
APA AppPerf   - Application Performance Analyser (NEW)
FAU FaultAn   - Fault Analyser (NEW)
 F FILEMANAGER - File Manager for Z/OS and OS/390
C5 CP/SM      - CICSplex/SM
 D Debug Tool - Debug Tool Utility functions
 G GIX        - Netview/DM Generalized Interactive Executive
 I ICSF       - Integrated Cryptographic Service Facility
LV Logrec View - Logrec Viewer
 M MQSERIES   - MQ Series panels (access authorisation needed)
 R RRS/ISPF   - Resource Recovery Panels
 T TPNS/ISPF  - TPNS V350
 W WLM        - Workload manager
    
```

```

Menu  Options  View  Utilities  Compilers  Help
-----
DSLIST - Data Sets Matching WRIGHTA.MV20.*                               Row 1 of 1
Command - Enter "/" to select action                                     Message          Volume
-----
fa  WRIGHTA.MV20.DBAKZCCA.D090127.T135402.S00214                       P1SD98
***** End of Data Set list *****
    
```

Fault Entry List

- **Fault History File**
 - Holds dump data
- **For System dumps**
- **For Transaction dumps too**
- **In reverse order**
 - Newest to oldest
- **Jobname**
 - For SDUMPs
- **Transid**
 - For tran dumps

```

IBM Fault Analyzer - Fault Entry List

Fault History File or View : 'PP.FAULTANL.HIST'

{The following line commands are available: ? (Query), V or S (View saved
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

  Fault ID Job/Tran User ID Sys/Job Abend Date      Time
-----
  F00034 COMKZCFM WARDJ  MV20  AP0001 2009/01/27 14:35:46
  F00032 TRP1      WRIGHTA DBAKZCCA ASRA  2009/01/27 13:54:04
  F00036 DBAKZCCA WRIGHTA  MV20  AP0001 2009/01/27 13:54:03
  F00030 TRP1      WRIGHTA DBAKZCCA ASRA  2009/01/27 10:12:33
  F00031 DBAKZCCA WRIGHTA  MV20  AP0001 2009/01/27 10:12:33
  F00025 TRA2      LEWISR  SSYKZCCL ASRA  2009/01/26 13:48:37
  F00024 TRA1      LEWISR  SSYKZCCL ASRA  2009/01/26 13:34:07
  F00022 GA71CTGS SKNIGHT MV24   U0000  2008/09/30 15:32:48
  F00020 CTGCRECO SMITHSO MV24   U4038  2008/08/13 15:31:56
  F00017 BOSKZCFL LEEPR   MV20   SM0002 2008/07/07 15:28:28
  F00016 DBAKZCEQ TAYLORM MV20   AP0001 2008/05/23 11:46:48

** Bottom of data.
    
```

The Query option - ?

- Query gives you a view of a Fault's info
- Each Fault has specific attributes:
 - abcode, Jobname, etc.
- Transaction dumps have info on transid, tasknum, etc
- Query provides an overview of a Fault
 - Helps locate the one of interest

```

File View Services Help
-----
Fault Entry Information

Fault ID. . . . . : F00032
User Name . . . . . : _____
User Title. . . . . : _____
Lock Flag . . . . . : ___ (Not locked)
Abend Code. . . . . : ASRA
POF Module Name . . . . . : PROGA1
POF Program Name. . . . . : PROGA1
POF Offset. . . . . : 92
Abend Date. . . . . : 2009/01/27
Abend Time. . . . . : 13:54:04
Job Name. . . . . : DBAKZCCA
Job ID. . . . . : JOB00850
Job Execution Class . . . . . : A
Job Type. . . . . : CICS
Job Step Name . . . . . : n/a
EXEC Program Name . . . . . : DFHSIP
User ID . . . . . : WRIGHTA
Group ID. . . . . : TSUSER
System Name . . . . . : MV20
Application ID. . . . . : IYCKZCCA
CICS Transaction ID . . . . . : TRP1
CICS Task Number. . . . . : 00033
Command ==> █
    
```

Analyzing a transaction dump Fault

- The dump can be interactively analysed (using I)
- The original report can be reviewed (using V or S)
- This was produced at the time of the transaction abend or **DUMP TRANSACTION** command

```

IBM Fault Analyzer - Fault Entry List

Fault History File or View : 'PP.FAULTANL.HIST'

{The following line commands are available: ? (Query), V or S (View saved
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

  Fault ID Job/Tran User ID Sys/Job Abend Date Time
  ---
  i F00034 COMKZCFM WARDJ MV20 AP0001 2009/01/27 14:35:46
  F00032 TRP1 WRIGHTA DBAKZCCA ASRA 2009/01/27 13:54:04
  F00036 DBAKZCCA WRIGHTA MV20 AP0001 2009/01/27 13:54:03
  F00030 TRP1 WRIGHTA DBAKZCCA ASRA 2009/01/27 10:12:33
  F00031 DBAKZCCA WRIGHTA MV20 AP0001 2009/01/27 10:12:33
  F00025 TRA2 LEWISR SSYKZCCL ASRA 2009/01/26 13:48:37
  F00024 TRA1 LEWISR SSYKZCCL ASRA 2009/01/26 13:34:07
  F00022 GA71CTGS SKNIGHT MV24 U0000 2008/09/30 15:32:48
  F00020 CTGCRECO SMITHSO MV24 U4038 2008/08/13 15:31:56
  F00017 BOSKZCFL LEEPR MV20 SM0002 2008/07/07 15:28:28
  F00016 DBAKZCEQ TAYLORM MV20 AP0001 2008/05/23 11:46:48

** Bottom of data.
  
```


The Interactive Reanalysis Report - I

- Fault Summary **describes the failing environment**
- The options list allows you to review further information about the cause of the dump
- Let's select the dump Synopsis (1) and then the Event Summary (2) ...

```
Interactive Reanalysis Report
TRANID: TRP1      CICS ABEND: ASRA      MV20      2009/01/27  13:54:04

Fault Summary:
Module PROGA1, CSECT PROGA1, offset X'92': CICS abend ASRA .

Select one of the following options to access further fault information:
  1. Synopsis
  2. Event Summary
  3. CICS Information
  4. Storage Areas
  5. Language Environment Heap Analysis
  6. User
  7. Abend Job Information
  8. Fault Analyzer Options

{Fault Analyzer maximum storage allocated: 2.35 megabytes.
DeferredReport processing execution time was 1.87 seconds (0.32 seconds CPU)}

*** Bottom of data.
```

Synopsis and Event Summary options (1 and 2)

- Synopsis (outlines the type of abend, and where it occurred)
- Event Summary (gives the program call stack that led to the abend)
- Yellow is hypertext!

```

Synopsis
TRANID: TRP1          CICS ABEND: ASRA          MV20          2009/01/27  13:54:04

A CICS abend ASRA occurred in module PROGA1 CSECT PROGA1 at offset X'92'.

A program-interruption code 0001 (Operation Exception) is associated with this
abend and indicates that:

    An attempt was made to execute an instruction with an invalid operation code.

The abend was caused by an undetermined instruction.
    
```

```

Event Summary
TRANID: TRP1          CICS ABEND: ASRA          MV20          2009/01/27  13:54:04

{The following events are presented in chronological order.}

Event #  Type  Fail Point  Module Name  Program Name  EP Name  Event Location (*)  Loaded From
-----
1  Call  DFHAPLI  DFHAPLI1  n/a  P+2460  TGRP.CICS650.SDFHLOAD
2  Call  CEEPLPKA  n/a  CEECRINI  E+AF6  PP.ADLE370.ZOS180.SCEERUN
3  Call  CEEPLPKA  n/a  CEECRINV  E+480  PP.ADLE370.ZOS180.SCEERUN
4  Call  CEEEV010  n/a  CEEEV010  E+310  PP.ADLE370.ZOS180.SCEERUN
5  Call  IBMLIB1  n/a  IBMRPMIA  E+51E  PP.ADLE370.ZOS180.SCEERUN
6  Link  PROG1  PROG1  PROG1  S#11 P+F8 E+F0  WRIGHTA.CICS650.P1LIB
7  Call  DFHAPLI  DFHAPLI1  n/a  P+2460  TGRP.CICS650.SDFHLOAD
8  Call  CEEPLPKA  n/a  CEECRINI  E+AF6  PP.ADLE370.ZOS180.SCEERUN
9  Call  CEEPLPKA  n/a  CEECRINV  E+302  PP.ADLE370.ZOS180.SCEERUN
10 Link  PROGCCL  PROGCCL  PROGCCL  P+342 E+342  WRIGHTA.CICS650.P1LIB
11 Call  DFHAPLI  DFHAPLI1  n/a  P+2460  TGRP.CICS650.SDFHLOAD
12 Abend ASRA  *****  PROGA1  PROGA1  n/a  P+92  WRIGHTA.CICS650.P1LIB
    
```

Clicking on P+92, from Event Summary stack...

- This link takes you to the memory location associate with the program + offset x'92'
- This shows the offending piece of code (0000 in this example)
- Fault Analyzer is built around this dynamic content navigation ...

```

Dump Storage
TRANID: TRP1          CICS ABEND: ASRA          MV20          2009/01/27  13:54:04

Address  Offset  Hex                                     EBCDIC
-----  -
Module PROGA1 CSECT PROGA1 + X'92'
000C10BA          0000 4110D068 *          ....}*
000C10C0          +6   41E03183 1BFF4100 30CE90E0 100041E0 *.\.c.....\...\*
000C10D0          +16  317850E0 100C9680 100C58F0 316C0DEF *..&\..o...0.%..*
000C10E0          +26  4110D068 41E03192 50E01000 96801000 *..}\..k&\..o...*
000C10F0          +36  58F0316C 0DEFE3D9 C1D5E2C1 C3E3C9D6 *.0.%..TRANSACTION*
000C1100          +46  D540E3D9 C1F140C3 D6D4D7D3 C5E3C540 *N TRA1 COMPLETE *
000C1110          +56  40404040                                     *          *
    
```

The CICS Information option (3)

- You can review the task's CICS environment at the time of the transaction dump
- The CICS Control Blocks shows the EIB, TWA, CSA, OFL, CWA, TCA, TACB, EIS, EIUS and TCTTE
- Transaction Storage shows all USER24, USER31, etc, storage
- Let's review the Summarized CICS Trace (5) and CICS Trace Formatting (6) ...

```

CICS Information
TRANID: TRP1          CICS ABEND: ASRA

CICS Release. . . . . : 0650
Application ID. . . . . : IYCKZCCA
CICS Transaction ID . . . . . : TRP1
CICS Task Number. . . . . : 00033
CICS Terminal ID. . . . . : V112
CICS Terminal Netname . . . . . : PYKSV112

Select one of the following:
 1. CICS Control Blocks
 2. CICS Transaction Storage
 3. Last CICS 3270 Screen Buffer
 4. Last CICS 3270 Screen Buffer Hex
 5. Summarized CICS Trace
 6. CICS Trace Formatting
 7. CICS Recovery Manager
 8. CICS Levels, Commareas, and Channels

*** Bottom of data.

```

Summarized CICS Trace (5) ...

```

Summarized CICS Trace                                     Line 1 Col 1 132
TRANID: TRP1      CICS ABEND: ASRA                      MV20      2009/01/27 13:54:04

00033 QR    AP 1940 APLI  ENTRY START_PROGRAM          PROGCL,CEDF,FULLAPI,EXEC,NO,14DBDBC,00000000 , 00000000,2,NO      =00000
00033 QR    AP 00E1 EIP   ENTRY LINK                      0004,14F30ED0 .3.},08000E02 ....      =00000
  Called-from-address 15D1073A : Module PROGCL program PROGCL + X'342'
00033 QR    PG 1101 PGLE  ENTRY LINK_EXEC              PROGA1,NO,NO                                =00000
00033 QR    AP 1940 APLI  ENTRY ESTABLISH_LANGUAGE      PROGA1,000C1000,000C1000,000001E0,USER,BELOW,NOT_DEFINED,00000000,LIN =00000
00033 QR    AP 1940 APLI  ENTRY START_PROGRAM          PROGA1,CEDF,FULLAPI,EXEC,NO,14DBDE00,00000000 , 00000000,3,NO      =00000
00033 QR    AP 00E1 EIP   ENTRY READ                      0004,001008D8 ...Q,08000602 ....      =00000
  Called-from-address 000C10B8 : Module PROGA1 CSECT PROGA1 + X'90'
00033 QR    DD 0201 DDDI  ENTRY ADD_ENTRY              00000000 , 14E1FB20,0004B344,14E44D80,YES,DSN,TCOM.IYC      =00000
00033 QR    DU 0601 DUTM  EXIT INQUIRE_SYSTEM_DUMPCODE/EXCEPTION DUMPCODE_NOT_FOUND,0,0,,,      *=00000
00033 QR    DU 0601 DUTM  EXIT INQUIRE_SYSTEM_DUMPCODE/EXCEPTION DUMPCODE_NOT_FOUND,0,0,,,      *=00000
00033 QR    AP F600 TDA   ENTRY WRITE_TRANSIENT_DATA  CSMT,13F678C0 , 00000002,NO      =00001
00033 QR    DU 0601 DUTM  EXIT INQUIRE_SYSTEM_DUMPCODE/EXCEPTION DUMPCODE_NOT_FOUND,0,0,,,      *=00001
00033 QR    AP F600 TDA   ENTRY WRITE_TRANSIENT_DATA  CSMT,14D22ABC , 00000001,NO      =00001
00033 QR    AP 00E1 EIP   EXIT READ                      OK                      00F4,00000000 ....,00000602 ....      *=00001

00033 QR    AP 0790 SRP   *EXC* PROGRAM_CHECK                                =00001
    
```

- **EXEC CICS commands are hyperlinked where they were issued**
 - Identifies calling program and offset within it
- **The Summarized CICS Trace defaults to the most interesting entries**
- **Standard Abbreviated trace formatting by default ...**

CICS Trace Formatting (6) ...

- Formatting options can be changed
- Short or Full trace may be used instead
- The Highlight Interval time period can be changed
- Can show more than summarized entries

```

CICS Trace Selection Parameters

Specify CICS trace selection parameters and press Enter.

Format . . . . . A (Abbrev/Short/Full)
Exception Only . . N (Yes/No)
Sequence Start . . 0000001
                  End . . . 0000433
Highlight Interval 0.128 (0-99.9999999999 secs)
Task IDs . . . . . _____
KE Task Numbers _____
Terminal IDs . . . _____ Caps Y
Transaction IDs _____ Caps Y
Time Start . . . . _____ (HHMMSS)
                  End . . . . _____ (HHMMSS)
Domain/Point IDs _____
    
```

CICS Levels, Commareas, Channels (8)

- **The CICS Link Level summary shows the nested hierarchy of the programs within the transaction being analysed**
 - PROGA1 (the program which had the abend) is at Link Level 3
 - Note that IDIXCX53 is seen because it was part of the Fault Analyzer run-time code which captured the transaction dump itself!
- **Clicking on an Event number takes you to a summary for that event (let's try number 12 ...)**
 - Events were also accessible from the **Event Summary (2)** ...

```

CICS Levels, Commareas, and Channels
TRANID: TRP1          CICS ABEND: ASRA

Number of Link Levels . . . : 4

Level 1 of 4 : Program PROGP1

Fault Analyzer Event #. . . : 6

Level 2 of 4 : Program PROGCCL

Fault Analyzer Event #. . . : 10

Level 3 of 4 : CSECT PROGA1

Fault Analyzer Event #. . . : 12

Level 4 of 4 : Load Module IDIXCX53

Fault Analyzer Event #. . . : n/a

Environment . . . . . : GLUE
  
```

CICS Event Details example

- **The abend Event, 12**
- **Not shown: instructions near point of failure, PSW, GPRs, EIB ...**
- **Older Event data relates to the linking programs**
- **No listing was available here**

```

Event 12 of 12: Abend ASRA *** Point of Failure ***
TRANID: TRP1      CICS ABEND: ASRA      MV20      2009/01/27  13:54:04

Previous Event Details

CICS Abend Code . . . . . : ASRA
Program-Interruption Code . : 0001 (Operation Exception)
  An attempt was made to execute an instruction with an invalid operation code.

NOTE: Source code information for CSECT PROGA1 could not be presented because
      no compiler listing or side-file data sets were provided.

Load Module Name. . . . . : WRIGHTA.CICS650.P1LIB (PROGA1)
  At Address. . . . . : 000C1000
  Load Module Length. . . . : X'1E0'
  Link-Edit Date and Time . : 2009/01/27  13:50:50

CSECT Name. . . . . : PROGA1
  At Address. . . . . : 000C1028 (Module PROGA1 offset X'28')
  CSECT Length. . . . . : X'19C'
  CSECT Language. . . . . : Assembler (Compiled using High Level Assembler

```


Relating an Event to the line of source

- **An example of when a program listing was available.....**
 - The assembly JCL had ADATA on its ASM PARM, and a step to IDILANGX to copy the ADATA metadata into a data set for Fault Analyzer to refer to

- **(Click on the List Stmt # to see failing instructions in context...**

```

TRANID: TRA1      CICS ABEND: ASRA      MV20      2009/02/05 14:27:11
CICS Abend Code . . . . . : ASRA
Program-Interruption Code . : 0001 (Operation Exception)
  An attempt was made to execute an instruction with an invalid operation code.

Assembler Source Code:
List
  Stmt #
000345 *****
000346 BANG    DS  H'0'

The IDILANGX file used for the above was found in WRIGHTA.LANGX(PROGA1).

Load Module Name. . . . . : WRIGHTA.CICS650.P1LIB(PROGA1)
  At Address. . . . . : 000C1000
  Load Module Length. . . . : X'1E0'
  Link-Edit Date and Time . : 2009/02/05 14:03:03

CSECT Name. . . . . : PROGA1
  At Address. . . . . : 000C1028 (Module PROGA1 offset X'28')
```

Seeing a failing source line in context

- Line 346 is the cause of the program check SOC1
- This is a very contrived example, to demonstrate how you can easily review your program source as part of dump analysis ...

```

CSECT PROGA1 Compiler Listing
TRANID: TRA1      CICS ABEND: ASRA      MV20      2009/02/05  14:27:11
      00000080 41F0 313C      LA      R15,316(,R3)
000341 +      STM      14,15,12(1)
      00000084 90EF 100C      STM      R14,R15,12(R1)
000342 +      OI      16(1),X'80'      LAST ARGUMENT
      00000088 9680 1010      OI      16(R1),128
000343 +      L      15,=V(DFHEI1)
      0000008C 58F0 316C      L      R15,364(,R3)
000344 +      BASR     14,15      INVOKE EXEC INTERFACE      @P7C
      00000090 0DEF      BASR     R14,R15
000345 *****
000346 BANG      DS      H'0'
      00000092 0000
000347 *      EXEC CICS SEND TEXT FROM(AREA) FREEKB
000348      DFHECALL =X'180660000800C20000082004000020',,(____RF,AREA),(*
000348      FB_2,=Y(L'AREA))
000349 *****
000350 +      DS      0H
000351 +      LA      1,DFHEIPL
      00000094 4110 D068      LA      R1,104(,R13)
    
```

Remaining Interactive Reanalysis Report options

- **4 - Storage Areas**
 - Shows Hex-dumped storage for the Events
- **5 - Language Environment Heap Analysis**
 - Breaks down the Heap storage used by the LE enclave
 - Bytes allocated, bytes free, number of storage elements
- **6 – User**
 - Shows the CICS concatenated data sets with source DSECTs
- **7 – Abend Job Information**
 - Shows the CICS job environment, DFHRPL and STEPLIB members, transaction program details and linkedit mapping, execution environment and LE runtime options
- **8 – Fault Analyzer Options**
 - Options in effect, data sets used by Fault Analyzer, exit programs invoked....

Abend Job Information (7) – Job Overview

- This provides a useful overview of the failing environment ...
- The job ID, Name, Step Name, ASID number, etc ...
- DFHRPL and STEPLIB are given (not shown here)

```

Abend Job Information
TRANID: TRP1      CICS ABEND: ASRA      MV20      2009/01/27 13:54:04

IBM Fault Analyzer Abend Job Information:

Abend Date. . . . . : 2009/01/27
Abend Time. . . . . : 13:54:04
System Name . . . . . : MV20
Job Type. . . . . : CICS Transaction
Job ID. . . . . : JOB00850
Job Name. . . . . : DBAKZCCA
Job Step Name . . . . . : CICS
ASID. . . . . : 53
Job Execution Class . . . . . : A
Region Size . . . . . : 0M
EXEC Program Name . . . . . : DFHSIP
User ID . . . . . : WRIGHTA
Accounting Information. . . . . : H251620-TS00
    
```

Abend Job Info (7) – Event-Related Programs

- The transaction's programs are listed, with their compilation and linkedit dates and times
- The failing program's linkedit map is expanded
- The CSECT AMODEs and RMODEs are displayed

Abend Job Information

TRANID: TRP1 CICS ABEND: ASRA MV20 2009/01/27 13:54:04

Event-Related Application Programs:

{The following list of event-related application programs is sorted by module link-edit date/time and program compilation date/time in reverse chronological order.}

Module Name	Link-Edit Date	Link-Edit Time	Program Name	Compilation Date	Compilation Time
PROGA1	2009/01/27	13:50:50	PROGA1	2009/01/27	n/a
PROGCCL	2009/01/27	11:46:51	PROGCCL	2009/01/27	11:46:49
PROGP1	2009/01/27	11:46:47	PROGP1	2009/01/27	11:46:46

Point Of Failure LINKEDIT Map:

Address	Offset	Length	Type	Date	Time	RMODE	AMODE	Language	Name
000C1000	0	1E0	MODULE	2009/01/27	13:50:50		24		PROGA1
000C1000	0	26	CSECT	2008/04/07		ANY	MIN	ASM	DFHEAT
000C1028	28	19C	CSECT	2009/01/27		24	24	ASM	PROGA1
000C11C8	1C8	16	CSECT	2008/04/07		ANY	MIN	ASM	DFHEAT0

Abend Job Info (7) – Environment / LE options

- **The z/OS level, JES type and CPU Model are returned**
 - Levels of SMS and LE are returned too

- **The Language Environment run-time options are formatted out**
 - Not all shown here for clarity
 - This gives a helpful overview of Language Environment settings

```

Abend Job Information
TRANID: TRP1      CICS ABEND: ASRA      MV20      2009/01/27 13:54:04
Execution Environment:

Operating System. . . . . : z/OS V1R8M0
Data Facility Product . . : DFSMS z/OS V1R8M0
Job Entry Subsystem . . . : JES2
Language Environment. . . : V1 R8.0
CPU Model . . . . . : 2094

Language Environment Run-Time Options:

Last Where Set      Option
-----
Installation default  ABPERC(NONE)
Installation default  ABTERMENC(ABEND)
Installation default  NOAIXBLD
Installation default  ALL31(ON)
Installation default  ANYHEAP(4096,4080,ANYWHERE,FREE)
Installation default  NOAUTOTASK
Installation default  BELOWHEAP(4096,4080,FREE)
Installation default  CBLOPTS(ON)
Installation default  CBLPSHPOP(ON)
Installation default  CBLQDA(OFF)
    
```

A SHOW storage example

- Enter SHOW command followed by an address
 - e.g. SHOW 1511C3E0
- The storage is displayed
- PF7 and PF8 to navigate backwards and forwards
- Relative offsets given

```

Dump Storage
SYSTEM=IYCKZCCA CODE=AP0001 ID=1/0001 MV20 2009/01/27 13:54:03

Address  Offset  Hex                                     EBCDIC
-----  -
File Control Table Entry (FCTE)  FILEA
1511C3E0          C6C9D3C5 C1404040 00000000 00000000 *FILEA .....*
1511C3F0      +10  00000000 01D8BA0A 80034406 84000000 *. ....Q.....d...*
1511C400      +20  00000000 00000001 00400000 00000000 *. ....*
1511C410      +30  00000000 00000000 00000000 00000004 *. ....*
1511C420      +40  0000000D 00000000 00000000 00000000 *. ....*
1511C430      +50  00000000 C3A877A4 0210E09E 00000000 *. ...Cy.u.\.....*
1511C440      +60  14E1FB20 14E1FB20 14E29000 13F61CB0 *. ....S...6...*
1511C450      +70  00000000 13FE29A0 C3A877A0 00000000 *. ....Cy.....*
1511C460      +80  00000000 00000000 00000000 00000000 *. ....*
      Lines 1511C470-1511C490 same as above
1511C4A0      +C0  0401A804 01000000 00000000 00010001 *. .y.....*
1511C4B0      +D0  00000000 00000000 00000000 00000000 *. ....*
1511C4C0      +E0  00000000 00000000 00000001 00000050 *. ....&*
1511C4D0      +F0  00000000 00020001 14E0B080 00000000 *. ....\.....*
1511C4E0      +100 00000000 00000000 40404040 40404040 *. ....*
1511C4F0      +110 00000000 00000000 00000000 00000000 *. ....*
      Lines 1511C500-1511C530 same as above
1511C540      +160 00000000 40404040 40404040 40404040 *. ....*
1511C550      +170 40404040 00000000 00000000 00000000 *. ....*
    
```

A DSECT mapping example

```

DSECT mapping for DFHEIUS at address 14DEE008                                     Line 1 Col
SYSTEM=IYCKZCCA CODE=AP0001 ID=1/0001 MV20 2009/01/27 13:54:03

14DEE008 +0000 DSECT DFHEIUS
EIOUS_START DS OA
EIOUS_PREFIX DS OCL12 Standard control block prefix
14DEE008 +0000 00B4 EIOUS_LENGTH DS H Length of DFHEIUS
14DEE00A +0002 6E EIOUS_ARROW DS C '>'
14DEE00B +0003 C4C6C8 EIOUS_DFH DS CL3 'DFH'
14DEE00E +0006 C5C9E4E2 40404040 4040 EIOUS_BLOCK_NAME DS CL10 'EIOUS'
14DEE018 +0010 00000000 EIOUS_CEE_TWA DS A Addr LE/370 Thread w/a @02C
EIOUS_STACK_AREA DS OA The whole link stack area
EIOUS_STACK_INIT DS OA Reinitialised section
14DEE01C +0014 00000000 00000000 EIOUS_CEE_RUNUNIT_TK DS CL8 CEE rununit token
14DEE024 +001C 00000000 DS A Reserved
14DEE028 +0020 00000000 DS A Reserved @P5A
EIOUS_INIT_LEN EQU *-EIOUS_STACK_INIT Length cleared
EIOUS_STACK_ASIS DS OA Left asis on the stack
EIOUS_CII_ARG_LIST DS OA COBOL II argument list
14DEE02C +0024 00000000 EIOUS_CII_ARG1 DS A COBOL II first argument
14DEE030 +0028 00000000 EIOUS_CII_ARG2 DS A COBOL II second argument
14DEE034 +002C 00000000 EIOUS_CII_ARG3 DS A COBOL II third argument
14DEE038 +0030 00000000 EIOUS_CII_ARG4 DS A COBOL II forth argument
14DEE03C +0034 00000000 EIOUS_CII_ARG5 DS A COBOL II fifth argument

```

- Select a control block address (e.g. DFHEIUS); press PF4
- Enter the DSECT name, and the storage is mapped for you ...

Analyzing a system dump Fault

- **As with transaction dumps, the system dump Fault can be interactively analyzed (using I)**
- **Remember the Fault can be created by running Fault Analyzer against the original SDUMP data set**
- **This example is the system dump that was generated by CICS at the time of the S0C1 which produced the ASRA transaction dump ...**

IBM Fault Analyzer - Fault Entry List

Fault History File or View : 'PP.FAULTANL.HIST'

{The following line commands are available: ? (Query), V or S (View saved report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H (Duplicate history).}

	<u>Fault ID</u>	<u>Job/Tran</u>	<u>User ID</u>	<u>Sys/Job</u>	<u>Abend</u>	<u>Date</u>	<u>Time</u>
—	F00039	TRA2	LEWISR	SSYKZCCL	ASRA	2009/01/29	09:20:00
—	F00038	TRA1	LEWISR	SSYKZCCL	AEI1	2009/01/29	09:18:04
—	F00037	TRA1	LEWISR	SSYKZCCL	ASRA	2009/01/28	16:07:54
—	F00034	COMKZCFM	WARDJ	MV20	AP0001	2009/01/27	14:35:46
—	F00032	TRP1	WRIGHTA	DBAKZCCA	ASRA	2009/01/27	13:54:04
i	F00036	DBAKZCCA	WRIGHTA	MV20	AP0001	2009/01/27	13:54:03
—	F00030	TRP1	WRIGHTA	DBAKZCCA	ASRA	2009/01/27	10:12:33
—	F00031	DBAKZCCA	WRIGHTA	MV20	AP0001	2009/01/27	10:12:33
—	F00025	TRA2	LEWISR	SSYKZCCL	ASRA	2009/01/26	13:48:37
—	F00024	TRA1	LEWISR	SSYKZCCL	ASRA	2009/01/26	13:34:07
—	F00022	GAT1CTGS	SKNIGHT	MV24	U0000	2008/09/30	15:32:48
—	F00020	CTGCRECD	SMITHSO	MV24	U4038		
—	F00017	BOSKZCFL	LEEPR	MV20	SM0002		
—	F00016	DBAKZCEQ	TAYLORN	MV20	AP0001		

Analyzing fault. Please wait...

CICS Transaction Dumps

- **Controlling transaction dumps**
 - ▶ via the CICS Transaction Dump Table
- **Printing CICS transaction dumps**
 - ▶ via the supplied dump utilities

The Transaction Dump Table

This is a dynamically created table used in CICS to control transaction dumps.

```
CEMT INQ TRDUMPCODE
```

```
STATUS:  RESULTS - OVERTYPE TO MODIFY
Trd(AEIN) Tra          Loc Max( 999 ) Cur(0003)
Trd(AEYD) Tra          Loc Max( 005 ) Cur(0001)
Trd(ARCH)           Sys          Loc Max( 002 ) Cur(0002)
Trd(ASRA) Tra          Loc Max( 010 ) Cur(0003)
Trd(ATNI) Tra          Loc Max( 000 ) Cur(0047)
Trd(GONE)           Shu Loc Max( 001 ) Cur(0000)
Trd(MAGA) Tra Sys      Loc Max( 999 ) Cur(0005)
Trd(SVIO) Tra Sys Shu Rel Max( 999 ) Cur(0000)
```

Dump related messages go to the CDUL log.

TRDUMAX (SIT option) controls the initial setting for the MAX parameter

Printing CICS Transaction Dumps

- **Refer to the CICS Operations and Utilities Guide for additional information.**
- **The CICS supplied Dump Utility Program**
 - ▶ DFHDU660 - CICS TS 4.1
 - ▶ DFHDU650 - CICS TS 3.2
 - ▶ DFHDU640 - CICS TS 3.1
 - ▶ DFHDU630 - CICS TS 2.3
- **The utility suffix represents the release number of CICS**
 - ▶ Note that CICS TS 3.1 used R640!

Printing CICS Transaction Dumps (continued)

- To list the available dumps in the transaction dump data set:

```
SELECT TYPE=SCAN
```

- Selecting dumps to be formatted by the utility:

```
SELECT TYPE={OR | NOTOR | AND | NOTAND | SCAN}
```

```
TRANID=({value|generic-value,...})
```

```
DUMPCODE=({value|generic-value,...})
```

```
DUMPID=({value|value-range,...})
```

```
PAGESIZE=({value|60})
```

```
TIME=({time|time-range,...})
```

```
UPPERCASE=(YES|NO)
```

```
END
```

- Example:

```
SELECT TYPE=OR, DUMPID=(1/0005)
```

```
SELECT
```

```
TYPE=NOTOR, TRANID=(CE+T, CS*), DUMPCODE=(ATN*, AZI6)
```

References and further reading ...

- **CICS TS 3.2 InfoCenter**
 - ▶ <http://publib.boulder.ibm.com/infocenter/cicsts/v3r2>
- **Rational Developer for System z**
 - ▶ <http://www-306.ibm.com/software/awdtools/rdz/>
- **Debug Tool for System z**
 - ▶ <http://www-306.ibm.com/software/awdtools/debugtool/>
- **Problem determination tools**
 - ▶ <http://www-306.ibm.com/software/awdtools/deployment>
- **Fault Analyzer**
 - ▶ <http://publibfp.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/idiugi00/CCONTENTS>
- **CICSplex System Manager WUI**
 - ▶ http://publib.boulder.ibm.com/infocenter/cicsts/v3r2/topic/com.ibm.cics.ts.cpsmwui.doc/topics/eyuad_overview.html
- **CICS Explorer**
 - ▶ <http://www-306.ibm.com/software/htp/cics/explorer/>

Summary

- **CICS debugging made easier...!**
- **Follow the cookbooks**
- **Be sure you are aware of what level of COBOL the program was compiled with:**
 - ▶ 2TGT versus 3TGT
 - ▶ Correct utilities
 - ▶ Correct manuals
- **Observe EYEBALL information in dumps**
- **Use log messages (CEEMSG, JESMSG LG, MSGUSR, etc).**
- **Fault Analyzer is extremely powerful and very useful**
- **Take advantage of the dump tables and the dump utilities**
- **Good luck ☺**