CICS TS Debugging Essentials: Irlink Suspends

Ehimare Uiyoshioria
IBM CICS Support
ehimareuiy@ibm.com

Agenda

- Premise
- Problem scenario
- Gathering documentation
- Debug
- Summary
- Useful Links
- Q&A

PREMISE

Premise: Which region is my problem???

- Nowadays one CICS region isn't enough to handle all the work thrown at you. Most shops have multiple regions but instead of having them run independent of each other they work together to handle all the workload.

- One region may be a file owning region(FOR) one may be an application owning region (AOR) or one may also be Terminal owning region (TOR).

- These regions are connected with CICS MulitRegion Operation(MRO) also know as IRC(Interregion Communication). MRO enables CICS systems that are running in the same MVS image, or in the same MVS sysplex, to communicate with each other.

- The goal of this session is to be able to investigate task suspend due to a connected task in another region.

PROBLEM SCENARIO

Problem Scenario

Regions connected via MRO start suddenly slowing down and tasks begin to backup

Using CEMT (CEMT INQ TASK) in a region you can see many of the tasks in an IRLINK type wait.

These task are taking up maxtask slots, and the region goes MAXTASK, affecting normal throughput.

You need to quickly determine what is causing the tasks to wait and not complete in a timely manner

GATHERING DOCUMENTATION

Gathering Documentation

- System dumps and job output (i.e. CICS Joblog) of all connected regions involved
  - If there are too many connected regions, making this an unreasonable request, then start with a system dump only of the region where the IRLINK waits are seen. Debug of this dump will reveal what connected region you will also need a system dump from.
  - Dumps can be created via:
    - DUMP command issued from the Console
    - CEMT P SNAP command issued online within CICS

DEBUG

Debug – Time of dump

IP IEAVDUMP

```
-------- GENERAL DUMP INFORMATION FOLLOWS --------
DUMP TITLE:       CICS DUMP: SYSTEM=IYNX8760 CODE=MT0001   ID=1/0001
DUMP TYPE:        SVC DUMP OF Z/OS HBB77D0, SNAME MV23
DUMP TAKEN:       FEB 5 2026, 18:37:25 (LOCAL)
```

IP ST SYS

```
TIME OF DAY CLOCK: E23283F8 897730A7  02/05/2026 18:37:25.529459 local
TIME OF DAY CLOCK: E23283F8 897730A7  02/05/2026 18:37:25.529459 GMT
```

Debug – Investigating transactions

To begin investigating the transactions we start in the dispatcher domain with IPCS command: VERBX DFHPDXXX ‘DS’

```
KE_TASK  T S F P TT RESOURCE RESOURCE_NAME    W TIME OF      TIMEOUT      DTA      AD ATTACHER MD SUSPAREA XM_TXN_TOKEN
                    TYPE                       SUSPEND      DUE          (DSTSK)     TOKEN

182D7000 N S P N -  IRLINK   8T09>AAH         M 18:37:00.845      -       495B0200 XM 19B37A00 QR 7F2D8B10 19B37A000000125C
182CF000 N S P N -  IRLINK   8T09>AAI         M 18:37:01.185      -       495B0380 XM 19B37D00 QR 7F2D8B70 19B37D000000127C
182C4000 N S P N -  IRLINK   8T09>AAJ         M 18:37:01.503      -       495B0500 XM 19B47100 QR 7F2D8BD0 19B471000000129C
182BC000 N S P N -  IRLINK   8T09>AAA         M 18:37:17.102      -       495B0680 XM 19B47400 QR 7F2D8870 19B474000000131C
182B4000 N S P N -  ALLOCATE 8T09             S 18:37:02.462      -       495B0800 XM 19B47700 QR 495B0800 19B477000000133C
182AC000 N S P N -  ALLOCATE 8T09             S 18:37:02.699      -       495B0980 XM 19B47A00 QR 495B0980 19B47A000000135C
182A4000 N S P N -  ALLOCATE 8T09             S 18:37:02.917      -       495B0B00 XM 19B47D00 QR 495B0B00 19B47D000000137C
18317000 N R                                                              495B1500 XM 183ACD00 QR          183ACD000000138C
1830F000 N S P N -  IRLINK   8T09>AAB         M 18:36:58.211      -       495B1680 XM 19B1E700 QR 7F2D88D0 19B1E7000000113C
182FF000 N S P N -  IRLINK   8T09>AAC         M 18:36:59.044      -       495B1800 XM 19B1EA00 QR 7F2D8930 19B1EA000000115C
182F7000 N S P N -  IRLINK   8T09>AAD         M 18:36:59.424      -       495B1980 XM 19B1ED00 QR 7F2D8990 19B1ED000000117C
182EF000 N S P N -  IRLINK   8T09>AAE         M 18:36:59.793      -       495B1B00 XM 19B37100 QR 7F2D89F0 19B371000000119C
182E7000 N S P N -  IRLINK   8T09>AAF         M 18:37:00.155      -       495B1C80 XM 19B37400 QR 7F2D8A50 19B374000000121C
182DF000 N S P N -  IRLINK   8T09>AAG         M 18:37:00.507      -       495B1E00 XM 19B37700 QR 7F2D8AB0 19B377000000123C
```

(XXX corresponds to the release of CICS you are running. 730, 740, 750, etc.)

Debug – Investigating transactions

As a comparison, here is how the Dispatcher domain looks in Fault Analyzer:

DS_TOKEN	KE_TASK	T	S	F	P	TT	Resource Type	Resource Name	W	Time Of Suspend	Timeout Due	DTA (DSTSK)	AD	Attacher Token	M	SUSPAREA	Task Num	Tran
000E0002	489BF000	S	S	P	N	IN	USERWAIT	DFHQRCPU	S	18:37:06.092	18:42:06.092	16861B00	XM	183AC700	L8	16861B00	00037	CQRC
00120003	1687A000	S	S	P	N	IN	XMCHKWAT	DFHXMCHK	S	18:37:06.092	18:42:06.092	16861E00	XM	183BBD00	L8	16861E00	00028	CXMT
00820002	1847EB00	S	S	N	N	IN			M	18:37:06.616	18:38:06.616	16882200	XM	183ACA00	QR	184E8098	00041	CISP
00840003	168D4000	S	S	N	N	-	ICMIDNTE	DFHAPTIM	S	00:00:00.083	11:56:53.423	16882380	XM	16908100	QR	16882380	00007	CSSY
00860003	48935000	S	S	N	N	-	ZC	DFHZNAC1	S	18:04:42.563	11:56:53.423	16882500	XM	16907400	QR	16882500	00042	CSNE
00880002	18359000	S	S	N	N	-	EPECQEMT	EPSUSPND	M	20:41:24.154	11:56:53.423	16882680	XM	16907D00	EP	169A7090	00006	CEPM
008A0003	1838E000	S	S	N	N	-	ICEXPIRY	DFHAPTIX	S	17:41:27.164	11:56:53.423	16882800	XM	16908400	QR	16862D70	00008	CSSY
00900002	168BB000	S	S	N	N	IN	SHSYSTEM		S	18:36:34.635	18:37:34.635	16882C80	XM	183BB700	QR	16862B00	00033	CSHQ
00920002	4894D000	S	R									16882E00	AP	000C0900	CQ			
01020002	1688B000	S	S	N	N	-	MPDQEMW	MPSUSPND	M	20:41:24.136	11:56:53.423	168CB200	XM	16907700	L8	1696D10C	00004	CMPE
010A0002	168EC000	S	S	N	N	IN	LGHARTBT	LG_MGRST	S	18:34:51.874	18:44:51.874	168CB800	LG	D3C7C8C2	QR	168CB800		
010E0002	168FC000	S	S	N	N	-	SODOMAIN	SO_NOWORK	M	18:11:09.218	11:56:53.423	168CBB00	XM	16907A00	SL	16928230	00005	CSOL
01100002	168FD000	S	S	N	N	IN	MNSYSTEM		S	20:42:46.099	11:56:53.423	168CBC80	MN	0000000F	QR	16862950		
01120003	1839C000	S	S	N	N	-	IS_SCHED	IS_SCHDQ	S	20:41:25.495	11:56:53.423	168CBE00	XM	183AC100	QR	168628F0	00040	CISM
01840005	183FB100	S	S	N	N	-	FCCFQR		M	20:41:24.343	11:56:53.423	168FF380	XM	183BB100	QR	183DE828	00026	CFQR
01860003	16883000	S	S	N	N	-	IS_ERROR	IS_ERROQ	S	20:41:25.493	11:56:53.423	168FF500	XM	16908700	QR	168681A0	00039	CISE
02040002	48966000	S	S	N	N	IN	WAIT	DFHHCHK	S	18:14:45.750	18:44:45.750	4895D380	XM	183AC400	L8	16862EF0	00036	CHCK
02060002	4896E000	S	S	N	N	-	RRMSEXIT	NOTIFICATION	M	20:41:23.613	11:56:53.423	4895D500	RX	16BF6B64	QR	169460C0		

We commonly use both IPCS and Fault Analyzer when reading dumps

Debug – Investigating transactions

RESOURCE TYPE	RESOURCE_NAME	W	TIME OF SUSPEND	TIMEOUT DUE	DTA (DSTSK)	AD	ATTACHER TOKEN	MD	SUSPAREA	XM_TXN_TOKEN
IRLINK	8TO9>AAH	M	18:37:00.845	-	495B0200	XM	19B37A00	QR	7F2D8B10	19B37A000000125C
IRLINK	8TO9>AAI	M	18:37:01.185	-	495B0380	XM	19B37D00	QR	7F2D8B70	19B37D000000127C
IRLINK	8TO9>AAJ	M	18:37:01.503	-	495B0500	XM	19B47100	QR	7F2D8BD0	19B471000000129C
IRLINK	8TO9>AAA	M	18:37:17.102	-	495B0680	XM	19B47400	QR	7F2D8870	19B474000000131C
ALLOCATE	8TO9	S	18:37:02.462	-	495B0800	XM	19B47700	QR	495B0800	19B477000000133C
ALLOCATE	8TO9	S	18:37:02.699	-	495B0980	XM	19B47A00	QR	495B0980	19B47A000000135C
ALLOCATE	8TO9	S	18:37:02.917	-	495B0B00	XM	19B47D00	QR	495B0B00	19B47D000000137C
					495B1500	XM	183ACD00	QR		183ACD000000138C
IRLINK	8TO9>AAB	M	18:36:58.211	-	495B1680	XM	19B1E700	QR	7F2D88D0	19B1E7000000113C
IRLINK	8TO9>AAC	M	18:36:59.044	-	495B1800	XM	19B1EA00	QR	7F2D8930	19B1EA000000115C
IRLINK	8TO9>AAD	M	18:36:59.424	-	495B1980	XM	19B1ED00	QR	7F2D8990	19B1ED000000117C
IRLINK	8TO9>AAE	M	18:36:59.793	-	495B1B00	XM	19B37100	QR	7F2D89F0	19B371000000119C
IRLINK	8TO9>AAF	M	18:37:00.155	-	495B1C80	XM	19B37400	QR	7F2D8A50	19B374000000121C
IRLINK	8TO9>AAG	M	18:37:00.507	-	495B1E00	XM	19B37700	QR	7F2D8AB0	19B377000000123C

- Now we see that we have several transactions in IRLINK or ALLOCATE suspends.
- IRLINK transactions are waiting for a response from a connected task in another region. These own sessions that other transactions want .
- Allocate Transactions have attempted to get a session to another CICS region but all the sessions are in use by the IRLINKs
- We can pick several of the oldest task based on suspend time
- Note the information under the Resource_Name column. This indicates the Connection name and Termid of the associated Session
- Tran# 113
  8TO9>AAB
  (Connection 8TO9, Session >AAB)
- Tran# 115
  8TO9>AAC
  (Connection 8TO9, Session >AAC)

Debug – Find the connection

Format Terminal Control domain to find associated Connection and Terminal with IPCS command: VERBX DFHPDXXX ‘TCP’

Make use of the find command to find the associated System Entry representing Connection 8TO9

```
Command ===> F TCTSE.8TO9
```

```
TCTSE.8TO9  183C61F0 MRO TCT SYSTEM ENTRY
F8E3D6F9 D0000000 183C61EC 19B0E0E0  19B0E240 E220016C C9E8D5E7 F9F7F6F0  *8TO9}...../...\\..S S..%IYNX9760*
00000000 00000000 19B27030 19B24030  02000000 183F9060 00000000 00000000  *.............. ........-........*
00000000 00000003 00030000 000A003F  0004000A 00000000 00000004 00000002  *................................*
00000000 00000000 00000019 00000003  00000000 00000000 00000000 00000000  *................................*
00000000 00000000 00000000 00000000  FFFF0000 00000000 00000000 00000000  *................................*
00000000 00010000 183C63B0 00000000  0002000C 00000000 00000089 FFFF0001  *..........................i....*
E23283E2 23738BC8 00000003 00000000  0000000B 40000000 00000000 00000000  *S.cS...H.......... ..........*
00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
00000000 00000000 00000000 00000000  00000000 00000000 D4D9D6C5 40404040  *........................MROE    *
E1EBEA11 20E0B07C E1EBEA11 20E0B07C  E4E2C1E2 E2C3F840 0001F0F7 F6F0E232  *.....\.@.....\.@USASSC8 ..0760S.*
7E7B63A7 75A6E4E2 C1E2E2C3 F8400001  00010000 00000000 000A000A 183C63B0  *=#.x.wUSASSC8 .................*
1440E232 7E7B6409 474A0000                                                *. S.=#...¢..                   *
```

Tran# 113
8TO9>AAB
(Connection 8TO9, Session >AAB)

Tran# 115
8TO9>AAC
(Connection 8TO9, Session >AAC)

The System entry lets us know the Applid of the connected region is IYNX9760.

Debug – Find the connection

Next, we can use Terminal Entry for the associated sessions to find the connected sessions in the connected region

Make use of the find command to find the Terminal Entry representing Session >AAB

```
Command ===> F TCTTE.>AAB
```

```
TCTTE.>AAB 19B242F0 TCT TERMINAL ENTRY

  0000  6EC1C1C2 D1000004 19B4EC00 19B4EC00  169BF100 0000113C 00000000 00000000  *>AABJ.............1.............*
  0020  00000000 0C404040 C5D5E400 00000086  00000000 00000000 00000000 00000000  *.....   ENU....f.....................*
  0040  00000000 00000000 00000000 00000000  00000001 02930000 00000000 18404170  *.....................l....... ..*
  0060  00000000 00000000 00040000 19B12110  00000000 00000000 00000000 00000000  *.....................................*
```

A bit under this is the Nib Descriptor

```
TCTENIB.>AAB 19B12110 NIB DESCRIPTOR

  0000  00000070 00000000 19B242F0 00000000  00000000 00000000 00000000 00000000  *...........0.....................*
  0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *.....................................*
  0040  00001001 084CC1C1 F5404040 40046EC1  C1C20000 00000000 00000000 00000000  *.....<AA5    .>AAB.............*
  0060  00000000 00000000 00000000 00000000                                       *................              *
```

Tran# 113
8TO9>AAB
(Connection 8TO9, Session >AAB)

Tran# 115
8TO9>AAC
(Connection 8TO9, Session >AAC)

The Nib Descriptor tells us that the connected task has Session <AA5

Debug – Find the connection

We can do the same thing for Session >AAC:

```
Command ===> F TCTTE.>AAC
```

```
TCTTE.>AAC 19B245B0 TCT TERMINAL ENTRY

  0000  6EC1C1C3 D1000004 19B5E400 19B5E400  169BF800 0000115C 00000000 00000000  *>AACJ.....U...U...8....*........*
  0020  00000000 0C404040 C5D5E400 00000086  00000000 00000000 00000000 00000000  *.....   ENU....f.............*
  0040  00000000 00000000 00000000 00000000  00000001 02930000 00000000 184041C0  *.....................l....... .{*
  0060  00000000 00000000 00040000 19B12180  00000000 00000000 00000000 00000000  *................................*
```

```
TCTENIB.>AAC 19B12180 NIB DESCRIPTOR

  0000  00000070 00000000 19B245B0 00000000  00000000 00000000 00000000 00000000  *................................*
  0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
  0040  00001001 084CC1C1 F6404040 40046EC1  C1C30000 00000000 00000000 00000000  *.....<AA6    .>AAC............*
  0060  00000000 00000000 00000000 00000000                                       *................                *
```

Tran# 113
8TO9>AAB
(Connection 8TO9, Session >AAB)

Tran# 115
8TO9>AAC
(Connection 8TO9, Session >AAC)

The connected task in the paired region has Session <AA6

Debug – Find the connection

What we know so far:

- Many task in IYNX8760 were sitting in IRLINK suspends
- The two oldest transactions in the suspends were tran# 113 and 115
- Both transactions were waiting on a response from Connection 8TO9 which relates to Applid IYNX9760
- Session >AAB in IYNX8760 is paired with Session <AA5 in IYNX9760
- Session >AAC in IYNX8760 is paired with Session <AA6 in IYNX9760

Next Step: Find the associated Sessions within IYNX9760 and determine why the associated transaction not responding back to their connected transactions

Now let's look at a dump from the connected region

Debug – Time of dump

IP IEAVDUMP

```
-------- GENERAL DUMP INFORMATION FOLLOWS --------
DUMP TITLE:       CICS DUMP: SYSTEM=IYNX9760 CODE=MT0001   ID=1/0001
DUMP TYPE:        SVC DUMP OF Z/OS HBB77D0, SNAME MV23
DUMP TAKEN:       FEB 5 2026, 18:37:28 (LOCAL)
```

IP ST SYS

```
TIME OF DAY CLOCK: E23283FB 5F31DAA6  02/05/2026 18:37:28.502045 local
TIME OF DAY CLOCK: E23283FB 5F31DAA6  02/05/2026 18:37:28.502045 GMT
```

Debug – Find the connection

Format Terminal Control domain to find associated Connection and Terminal with IPCS command: VERBX DFHPDXXX ‘TCP’

Use the find command to find the Terminal Entry representing Session <AA5

```
Command ===> F TCTTE.<AA5
```

```
TCTTE.<AA5 19A342F0 TCT TERMINAL ENTRY

  0000  4CC1C1F5 D1000006 19A1DB20 19A1DB20  169BE800 0000106C 00000000 00000000  *<AA5J.............Y....%........*
  0020  00000000 0C404040 C5D5E400 00000086  00000000 00000000 00000000 00000000  *.....   ENU....f.................*
  0040  00000000 40000000 00000000 00000000  00000002 02930000 00000000 183DCAD0  *....  .................l.........}*
  0060  00000000 00000000 00000000 19A1AE30  00000000 00000000 00000000 00000000  *.................................*
```

Using the Nib Descriptor beneath, we can confirm the matched pair

```
TCTENIB.<AA5 19A1AE30 NIB DESCRIPTOR

  0000  00000070 00000000 19A342F0 00000000  00000000 00000000 00000000 00000000  *.........t.0.....................*
  0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *.................................*
  0040  00001001 044CC1C1 F5086EC1 C1C24040  40400000 00000000 00000000 00000000  *.....<AA5.>AAB    ..............*
  0060  00000000 00000000 00000000 00000000                                       *................              *
```

Debug – Find the connection

We can do the same thing for <AA6

```
Command ===> F TCTTE.<AA6
```

```
TCTTE.<AA6 19A345B0 TCT TERMINAL ENTRY

  0000  4CC1C1F6 D1000006 19A22000 19A22000  169C0100 0000107C 00000000 00000000  *<AA6J....s...s.........@........*
  0020  00000000 0C404040 C5D5E400 00000086  00000000 00000000 00000000 00000000  *.....   ENU....f.................*
  0040  00000000 40000000 00000000 00000000  00000002 02930000 00000000 183DCB20  *....  ................l.........*
  0060  00000000 00000000 00000000 19A1AEA0  00000000 00000000 00000000 00000000  *................................*
```

```
TCTENIB.<AA6 19A1AEA0 NIB DESCRIPTOR

  0000  00000070 00000000 19A345B0 00000000  00000000 00000000 00000000 00000000  *.........t.....................*
  0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
  0040  00001001 044CC1C1 F6086EC1 C1C34040  40400000 00000000 00000000 00000000  *.....<AA6.>AAC     ..............*
  0060  00000000 00000000 00000000 00000000                                       *................               *
```

Now the sessions are connected to task# 106 and 107.

Debug – Investigating transactions

Now lets Format the Kernel domain to find associated stack entries for the two tasks with IPCS command: VERBX DFHPDXXX ‘KE’

First find on the task# to get the KE_NUM

```
Command ===> F 0106
```

```
===KE: Kernel Domain KE_TASK Summary

KE_NUM KE_TASK  STATUS       TCA_ADDR TRAN_# TRANSID DS_TASK  KE_KTCB  ERROR    TCB

00F9   1830F000 Not Running  169C0100 00107  CSMI    495B1680 167DC2F0
00FA   18317000 Not Running  169BE800 00106  CSMI    495B1500 167DC2F0
00FB   18322000 ***Running** 169C4100 00116  CEMT    495B1200 167DC2F0        008D62A8
0101   183E8000 Not Running  169BB800 00027  CSNC    489A6E00 167DC2F0
0103   183F9000 Not Running  169BA800 00024  CEPF    168CBC80 48915F18
0104   183F9B00 Not Running  169BA100 00025  CFQS    168FF380 167DC2F0
0106   183FB100 KTCB EP001   00000000                168F3000 48915F18        008A5E88
```

Debug – Investigating transactions

Now that we have the KE_NUMs we can find on them to get down to their Kernel stack which shows the current sequence of processing for a task at the time the dump was taken:

```
KE_NUM @STACK   LEN  TYPE ADDRESS  LINK REG OFFSET ERR NAME

00FA   18318040 0200 Bot  16503000 9650365E 00065E     DFHKETA
00FA   18318240 03D0 Dom  16525AD0 96525D5A 00028A     DFHDSKE
00FA   18318610 1690 Dom  1657A0B0 9657C4FA 00244A     DFHXMTA
00FA   18319CA0 0460 Dom  172AAB00 972AB1FC 0006FC     DFHAPPG
                     Int   +000466 972AAC8A 00018A       FAST_INITIAL_LINK
00FA   1831A100 12B8 Lifo 189F5670 9729BAAA 000000     DFHMIR
00FA   1831B3B8 1610 Lifo 1729B900 9729D49A 001B9A     DFHEPC
00FA   1831C9C8 1420 Dom  16D00000 96D03C1E 003C1E     DFHPGLE
                     Int   +000B56 96D003EA 0003EA       LINK_EXEC
00FA   1831DDE8 10D0 Dom  17288000 800C5D70 000000     DFHAPLI1
                     Int   +004D8C 972892FC 0012FC       LE370_INTERFACE
                     Int   +004B2C 9728E3D2 0063D2       INVOKE_FOR_RECURSION
00FA   18279040 1BB0 Sub  17879F00 9787E79E 00489E     DFHEIIC
                     Int   +00469E 9787AA24 000B24       CALL_ICP
00FA   1827ABF0 0B40 Lifo 171D4F00 971D8E16 003F16     DFHICP
00FA   1827B730 0470 Dom  16516900 96518B5E 00225E     DFHDSSR
                     Int   +001982 965174CA 000BCA       POP_TASK
```

- For most non-system task, the first ~4 entries will be very similar as the represent the process of starting up a task.
- The Initial Link gives control to the initial program that will run on behalf of the task
- The LINK_EXEC represents an EXEC CICS Link to program (program name to be determined)
- We can also see a call to Interval Control program (DFHICP) via another EXEC CICS call
- And it ends with a call to Dispatcher (DFHDSSR) to suspend the task

Debug – Investigating transactions

Format the Dispatcher domain to view what dispatcher indicates the problem task are doing with IPCS command: VERBX DFHPDXXX ‘DS’

RESOURCE TYPE	RESOURCE_NAME	W	TIME OF SUSPEND	TIMEOUT DUE	DTA (DSTSK)	AD	ATTACHER TOKEN	MD	SUSPAREA	XM_TXN_TOKEN
TCP_NORM	DFHZDSP	W	18:37:28.275	-	4895DC80	XM	16908A00	QR	00042110	16908A000000010C
SMSYSTEM		S	18:34:14.125	18:39:14.125	489A6800	SM	00000002	QR	16862A70	
SMOM	Monitor_wait	M	18:37:14.480	18:38:14.480	489A6980	SM	00000023	QR	1684279C	
SMON SUS	SM z/OS notify	S	20:58:34.154	-	489A6B00	SM	00000024	QR	16862AA0	
RRMSEXIT	RESYNC	M	20:58:34.154	-	489A6C80		16BF6D2A	QR	169460F0	
CSNC	MROQUEUE	M	18:37:17.103	-	489A6E00	XM	183C3A00	QR	1845803C	183C3A000000027C
ICWAIT	<ABC	S	18:37:01.185	-	495B0200	XM	19A28A00	QR	495B0200	19A28A000000113C
ICWAIT	<ABD	S	18:37:01.503	-	495B0380	XM	19A28D00	QR	495B0380	19A28D000000114C
					495B1200	XM	16908700	QR		169087000000116C
ICWAIT	<AA4	S	18:37:17.103	-	495B1380	XM	183B1400	QR	495B1380	183B14000000115C
ICWAIT	<AA5	S	18:36:58.211	-	495B1500	XM	16907400	QR	495B1500	169074000000106C
ICWAIT	<AA6	S	18:36:59.044	-	495B1680	XM	184F0700	QR	495B1680	184F07000000107C
ICWAIT	<AA7	S	18:36:59.424	-	495B1800	XM	184F0A00	QR	495B1800	184F0A000000108C
ICWAIT	<AA8	S	18:36:59.793	-	495B1980	XM	184F0D00	QR	495B1980	184F0D000000109C
ICWAIT	<AA9	S	18:37:00.155	-	495B1B00	XM	19A28100	QR	495B1B00	19A281000000110C
ICWAIT	<ABA	S	18:37:00.507	-	495B1C80	XM	19A28400	QR	495B1C80	19A284000000111C
ICWAIT	<ABB	S	18:37:00.845	-	495B1E00	XM	19A28700	QR	495B1E00	19A287000000112C

This domain shows our two tasks suspended in an ICWAIT (Interval Control wait). Our goal is to determine the program that issued the wait. This program is responsible for the IRLINK waits on the connect region IYNX8760. Notice many other tasks are in ICWAIT as well.

Debug – Investigating transactions

Format the Application domain to find the associated Exec Interface control blocks with IPCS command: VERBX DFHPDXXX ‘AP’

There are 2 EIBs per task – a SYSTEM EIB and a USER EIB. Use the find command to find the SYSEIB for the task:

```
Command ===> F SYSEIB.00106
```

```
SYSEIB.00106 169BEB88 System EXEC Interface Block

 -0008                                                5CE2E8E2 C5C9C240  *                    *SYSEIB *

  0000  0183658C 0126036F D7D9C5C4 0000106C  4CC1C1F5 00000000 00000010 04000000  *.c.....?PRED...%<AA5............*
  0020  00000000 00000000 00000000 00000000  00000040 40404040 40404000 00000000  *.....................        .....*
  0040  00000000 00000000 00000000 00000000  00000000 00                         *.......................          *
```

EIBs contains the EIBFN(function code) representing the last EXEC CICS command the task used at **offset x’1B’.** It is a two byte field and currently contains x’1004’ which equates to an EXEC CICS DELAY.

Debug – Investigating transactions

When an EXEC CICS command completes the SYSEIB is copied to the USER EIB

Find the USER EIB for the task:

```
Command ===> F EIB.00106
```

```
EIB.00106 18520100 EXEC Interface Block

 -0010                                 00656EC4 C6C8C1D7 6DC4C6C8 C5C9C25C  *                ..>DFHAP_DFHEIB**

  0000  0183658C 0126036F D7D9C5C4 0000106C  4CC1C1F5 00000000 0000000E 02000000  *.c.....?PRED...%<AA5.............*
  0020  00000000 00000000 00000000 00000000  000000C4 C5D3C1E8 D4C54000 00000000  *....................DELAYME .....*
  0040  00000000 00000000 00000000 00000000  00000000 00                         *.....................          *
```

The EIBFN value is different as it currently contains x'0E02' which equates to an EXEC CICS LINK. This is the last EXEC CICS command that completed. This likely means the command represented in the SYSEIB is still processing.

Debug – Investigating transactions

Now we want to locate where the DELAY command was issued from. We can do this with the EIUS (EXEC Interface User Structure) which is located between the SYSEIB and EIB. The EIUS contains the applications register save area address at offset +x'3C'

```
EIUS.00106 18520008 EXEC Interface User Structure

  0000  00E86EC4 C6C8C5C9 E4E24040 40404040  18520878 00000000 18524600 00000000  *.Y>DFHEIUS      ...............*
  0020  00000000 00000000 00000000 00000000  00000000 18520100 00000000 18529DE0  *.............................\*
  0040  00000000 00000000 00000000 9728CC0E  00028968 1683E000 1831E3C8 1831E28C  *............p.....i..c\...TH..S.*
  0060  172905C0 1831DDE8 00080048 169BEA88  1831E4A8 7F4FF878 18317000 1831E518  *...{...Y.......h..Uy"|8.......V.*
  0080  183A79B0 169BE800 00000000 00000000  00000000 00000000 00000000 00000000  *......Y.........................*
  00A0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
  00C0  00000000 00000000 00000000 00000000  00000000 00000000 1852003C 18520040  *............................... *
  00E0  00000000 00000000                                                         *........                       *
```

After the command is processed CICS will restore the application register contents from this rsa before giving control back to the application.

Debug – Investigating transactions

An RSA has a set format as follows:

- Word 1 reserved RSA+’0’
- Word 2 Address of the save area used by the calling program (backward chain pointer) RSA+’4’
- Word 3 Address of the save area set up by the called program (forward chain pointer) RSA+’8’
- Word 4 Address to which to return (register 14) RSA+’C’
- Word 5 Address of the entry point (register 15) RSA+’10’
- Word 6 Contents of register 0 RSA+’14’
- Word 7 Contents of register 1 (typically this will be the address of the parameter list) RSA+’18’
- Word 8 Contents of register 2 RSA+’1C’
- Word 9 Contents of register 3 RSA+’20’
- Word 10 Contents of register 4 RSA+’24’
- Word 11 Contents of register 5 RSA+’28’
- Word 12 Contents of register 6 RSA+’2C’
- Word 13 Contents of register 7 RSA+’30’
- Word 14 Contents of register 8 RSA+’34’
- Word 15 Contents of register 9 RSA+’38’
- Word 16 Contents of register 10 RSA+’3C’
- Word 17 Contents of register 11 RSA+’40’
- Word 18 Contents of register 12 RSA+’44’

Debug – Investigating transactions

Within IPCS browse mode we can display the RSA address taken from EIUS+x'3C' (18529DE0)

```
18529DE0   00110301   18529C38   00000000 ? 99C018F6   | ............r{.6 |
18529DF0   00000000   00000000   18529ED8   9852B618   | ............Qq... |
18529E00   19C01C68   00000010   18529DE0   1831E408   | .{.........\..U. |
18529E10   00000000   1852B490   1852B578   19C016C4   | ............{.D  |
18529E20   1852003C   18528F08   00000000   18529FE8   | ................Y_|
```

Based on the previous slide RSA+'C' holds register 14 which is the return address. This will point within the program that issues the EXEC CICS DELAY

R14-99C018F6

- You will need to take off the high order "80" bit, so the 31-bit address is 19C018F6
- In browse mode display this address
- Back up 2 bytes with command: L -2

```
19C018F4                0DEF40F0   9008E54C   D0F00000   |     .. 0..V<}0.. |
```

- We find instruction 0DEF which is a BASR where the application gives up control to CICS to process the EXEC CICS command

Debug – Investigating transactions

Using this address in R14 (19C018F6) to identify the program that issued the command.

You can page backwards in browse mode looking for an eye catcher like “CEE”

Backing up we can find eye catcher CEE located at 19C016A0

```
19C016A0   47F0F014   01C3C5C5   00000208   000006B8   | .00..CEE........ |
```

This means the program name is further down in storage by the length of the value in offset +’C’

We can use command: L +6B8

```
19C01D58                         20CEA506   000008A0   |         ..v..... |
19C01D60   00000810   00000000   FFFF0000   40000000   | ............ ... |
19C01D70   90000208   00020012   D0000190   00000000   | ........}....... |
19C01D80.:19C01D8F. LENGTH(X'10')--All bytes contain X'00'
19C01D90   00000000   80000000   0007C4C5   D3C1E8D4   | .........DELAYM |
19C01DA0   C5000000   00000000   20CE2106   000005B8   | E............... |
```

Notice the name DELAYME, this is likely the program name that issued the EXEC CICS DELAY

Debug – Investigating transactions

We can verify this in the Loader Domain with IPCS command: VERBX DFHPDXXX ‘LD’

Then 'Find' Using the first 3 digits of the return address (99C018F6) like so:

```
Command ===> F '99C'
```

```
PGM NAME ENTRY PT  CSECT    LOAD PT.
PREDELAY 99C00000 -noheda- 19C00000
DELAYME  99C016A0 -noheda- 19C016A0
```

Focusing on the LOAD PT. column, determine where the 31 bit return address falls inside of. In this case it is DELAYME.

You can subtract the entry pt from R14 address to get the offset into the program where the command was issued: 99C018F6-99C016A0= 256

The delay command was issued from offset x’256’ into program DELAYME

```
SYSEIB.00107 169C0488 System EXEC Interface Block

 -0008                                                 5CE2E8E2 C5C9C240  *                    *SYSEIB *

  0000  0183659C 0126036F D7D9C5C4 0000107C  4CC1C1F6 00000000 00000010 04000000  *.c.....?PRED...@<AA6.............*
  0020  00000000 00000000 00000000 00000000  00000040 40404040 40404000 00000000  *....................        .....*
  0040  00000000 00000000 00000000 00000000  00000000 00                         *......................              *
```

```
EIB.00107 18530100 EXEC Interface Block

  -0010                                  00656EC4 C6C8C1D7 6DC4C6C8 C5C9C25C  *              ..>DFHAP_DFHEIB**

   0000  0183659C 0126036F D7D9C5C4 0000107C  4CC1C1F6 00000000 0000000E 02000000  *.c.....?PRED...@<AA6.............*
   0020  00000000 00000000 00000000 00000000  000000C4 C5D3C1E8 D4C54000 00000000  *...................DELAYME .....*
   0040  00000000 00000000 00000000 00000000  00000000 00                         *......................              *
```

```
EIUS.00107 18530008 EXEC Interface User Structure

   0000  00E86EC4 C6C8C5C9 E4E24040 40404040  18530878 00000000 18534600 00000000  *.Y>DFHEIUS      ................*
   0020  00000000 00000000 00000000 00000000  00000000 18530100 00000000 18539DE0  *.............................\*
   0040  00000000 00000000 00000000 9728CC0E  00028968 1683E000 183163C8 1831628C  *............p.....i..c\....H....*
   0060  172905C0 18315DE8 00080048 169C0388  183164A8 7F4FF878 1830F000 18316518  *...{..)Y.......h...y"|8...0.....*
   0080  183A79B0 169C0100 00000000 00000000  00000000 00000000 00000000 00000000  *.............................*
   00A0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *.............................*
   00C0  00000000 00000000 00000000 00000000  00000000 00000000 1853003C 18530040  *............................ *
   00E0  00000000 00000000                                                        *........                     *
```

```
18539DE0   00110301   18539C38   00000000   99C018F6   | ............r{.6 |
18539DF0   00000000   00000000   18539ED8   9853B618   | ............Qq... |
18539E00   19C01C68   00000010   18539DE0   18316408   | .{.........\.... |
18539E10   00000000   1853B490   1853B578   19C016C4   | ..............{.D |
18539E20   1853003C   18538F08   00000000   18539FE8   | ................Y |
```

Register 14 Is the same as in tran# 106

SUMMARY

Summary

- Many tasks on IYNX9760 are sitting in Interval Control waits (ICWAIT)
- Terminal (Session) <AA5 is running task number 106 (CSMI)
- Terminal (Session) <AA6 is running task number 107 (CSMI)
- Both tasks issued an EXEC CICS DELAY
- For both tasks program DELAYME is responsible for issuing the delay
- The delay is causing the associated task on the IYNX8760 to wait in IRLINK wait.
- Application programmer, responsible for program DELAYME, should be made aware of the cause and effect of these findings and take appropriate actions as necessary.

USEFUL LINKS

Useful Links for similar problems

- https://www.ibm.com/docs/en/cics-ts/6.x?topic=waits-investigating-terminal
- https://www.ibm.com/docs/en/cics-ts/6.x?topic=terminal-investigating-related-task-in-remote-region
- https://www.ibm.com/docs/en/cics-ts/6.x?topic=waits-resources-that-cics-tasks-can-wait

QUESTIONS?

Experience more with IBM

Visit us at the IBM Booth #113

After a full day of technical sessions, take a break with us!

Connect with our experts, snap a photo with the z17 Plexi or the latest Telum II, and get an up-close look at our Spyre Accelerator.

Come back each day for fresh topics and demos at our expert stations.

Think 2026

Join 5000+ senior business and technology leaders who are seizing the AI revolution to unlock unprecedented growth and productivity at Think 2026.

Find out more information using the QR code below.

IBM Digital Asset Haven

IBM Digital Asset Haven is the operational backbone for financial institutions and regulated enterprises entering the digital asset economy.

Find out more information using the QR code below.

Your feedback is important!

SHARE
EDUCATE • NETWORK • INFLUENCE

Submit a session evaluation for each session you attend:

www.share.org/evaluation