Opening the CICS Toolbox

Peggy DeVal
IBM CICS Support Center

Agenda

- Dump types
- Some basic IPCS commands
- CICS Verbexit DFHPDxxx
  - Common System Area (CSA)
  - Storage Manager (SM)
  - Transaction Manager (XM)
  - Dispatcher (DS)
  - Kernel (KE)
  - Loader domain (LD)
  - Program domain (PG)
  - Lock Manager (LM)
  - Enqueue domain (NQ)
  - Application domain (AP)
  - Trace (TR)

Dump types formatted with IPCS

- SVC dumps (also called SDUMPs).
  - Dumps requested by CICS (or occasionally another product).
    - Dumps requested by CICS will include CICS DUMP in the title:

      Dump Title: CICS DUMP: SYSTEM=IYNXS CODE=XM0002 ID=64/0064

  - SLIP dumps (dumps taken by SLIP traps with ACTION=SVCD).
  - Console dumps (requested via the MVS console).
    - Should typically specify SDATA parameters similar to the following when requesting a console dump:

      DUMP COMM=(dump title)
      /r nn,JOBNAME=(CICS jobname),CONT
      /r mm,SDATA=(ALLNUC,CSA,LPA,PSA,RGN,SQA,SUM,TRT),END

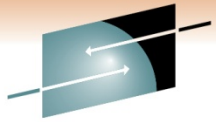
- Rarely in CICS, may also see:
  - SYSMDUMPs
  - Stand-Alone dumps

Some useful IPCS commands

- **CBFORMAT (CBF)** – Displays a formatted control block.
  - **CBF RTCT** – Formats the Recovery Termination Control Table.
    - Can be used to see what address spaces are included in the dump.
  - **CBF address STR(control block name)** e.g. **ip cbf 12345678 str(rtm2wa)**
- **EQUATE (EQ)** – Assigns a name to the current address.
- **FIND (F)** – Locate data in a dump.
  - **F 'charstring' NOB** (NOB indicates No Break for uninterrupted searching).
  - **F x'datastring' NOB** - Locates a hexadecimal string.
- **LIST (L)** – Display storage at an address.
  - **L addr LEN(x'1000')** – List x'1000' bytes starting at the specified address.
  - **L addr+100** – List the specified address +x'100'.
  - **L SLIPTRAP** – List the slip trap for which a SLIP dump was taken.
  - **L addr INSTR** – Displays Assembler instructions at the specified address.
    - e.g. **ip l 23d00 l(x'20') i**

```
LIST 023D00. ASID(X'0097') LENGTH(X'20') INSTRUCTION
ASID(X'0097') ADDRESS(023D00.) KEY(00) ABSOLUTE(49FD00.)
00023D00 | BF6F C2F4      | ICM   R6,X'F',X'2F4'(R12)
00023D04 | 9505 600A      | CLI   X'A'(R6),X'05'
00023D08 | 4770 B056      | BC    X'7',X'56'(,R11)
00023D10 | 58F0 6144      | L     R15,X'144'(,R6)
00023D14 | 0DEF           | BASR  R14,R15
```

Some useful IPCS commands (cont)



- **LISTTOD (LTOD)** – Convert TOD clock to corresponding timestamps.
  - e.g. **ip ltod C6BC0E414486E884**

```
10/15/2010 23:58:44.955246 STCK   X'C6BC0E41 4486E884'
10/15/2010 23:58:44.955246 UTC    X'C6BC0E41 4486E884'
10/15/2010 17:58:44.955246 LOCAL  X'C6BBBDC9 E706E884'
```

  - Top STCK is formatted GMT time without being adjusted for time zone or leap seconds.
  - Middle UTC has been adjusted for leap seconds.
  - Local time has been adjusted for both time zone and leap seconds.
- **HELP** – Provides a list of IPCS commands and subcommands.
  - **HELP** subcommand e.g. **ip help runchain**
- **LPAMAP** – Display LPA module summary.
- **OPCODE** – Converts a hex instruction to it's corresponding Assembler instruction.
- **RUNCHAIN (RUNC)** – Format a chain of control blocks.
  - e.g. **ip runc addr(12345678) len(x'20') link(x'10') chain(99) display verify**

ADDR - Address of first block in chain.
LEN - Length of storage to be displayed
LINK - Offset in each block for the next block on the chain.
CHAIN - Number of control blocks to chain; default is 999.

Commonly used IPCS commands (cont)

- **SELECT** – Displays address space/jobname information.
  - **ALL**
  - **ASID (x'…')** or **ASID (x'…',x'…')** for multiple address spaces.
- **STATUS (ST)** – Provides system status; includes dump title.
  - **STATUS SYSTEM (ST SYS)** – Includes time dump was taken.
  - **STATUS REGS (ST REGS)** – Formats PSW/registers when SLIP trap springs.
- **SUMMARY FORMAT (Summ Format)** – Formats TCBs and related information.
- **SYSTRACE** – Format MVS system trace table.
  - **TIME(LOCAL)**
  - **ALL** - Request that all address spaces be included.
  - e.g. **systrace time(local) asid(x'101') tcb(x'9F8220')**
- **VERBEXIT (VERBX)** – User-supplied VERB exits – CICS, DB2, LE/370 etc.
  - **DFHPD*xxx*** (where '*xxx*' is the CICS release e.g. 640, 650, 660).
  - **VERBX MTRACE** – Formats the most recent console messages.
  - **VERBX LOGDATA** – Formats the most recent LOGREC entries.
- **WHERE (W)** – Identify an area of storage.
  - e.g. **Where 0E200200**:

```
ASID(X'0097') 0E200200. AREA(Subpool252Key00)+0200 IN EXTENDED PRIVATE
ASID(X'0097') 0E200200. DFHSIP+0200 IN EXTENDED PRIVATE
```

IPCS Subcommand Menu


```
------------------------- IPCS Subcommand Entry ----------------------------
-
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation below:

===> verbx dfhpd660 'ke=1'


---------------------- IPCS Subcommands and Abbreviations ------------------
-
ADDDUMP            | DROPDUMP, DROPD    | LISTDUMP, LDMP     | RENUM,     REN
ANALYZE            | DROPMAP,  DROPM    | LISTMAP,  LMAP     | RUNCHAIN, RUNC
ARCHECK            | DROPSYM,  DROPS    | LISTSYM,  LSYM     | SCAN
ASCBEXIT, ASCBX    | EPTRACE            | LISTUCB,  LISTU    | SELECT
ASMCHECK, ASMK     | EQUATE,   EQU, EQ  | LITERAL            | SETDEF,    SETD
CBFORMAT, CBF      | FIND,     F        | LPAMAP             | STACK
CBSTAT             | FINDMOD,  FMOD     | MERGE              | STATUS,    ST
CLOSE              | FINDUCB,  FINDU    | NAME               | SUMMARY,   SUMM
COPYDDIR           | GTFTRACE, GTF      | NAMETOKN           | SYSTRACE
COPYDUMP           | INTEGER            | NOTE,     N        | TCBEXIT,   TCBX
COPYTRC            | IPCS HELP, H       | OPEN               | VERBEXIT, VERBX
CTRACE             | LIST,     L        | PROFILE,  PROF     | WHERE,     W
```

Status commands

- **Status (ST)**
  - Displays the dump title:

    Dump Title: CICS DUMP: SYSTEM=IYNXS CODE=XM0002 ID=64/0064

  - The title tells you who requested the dump:
    - CICS, SLIP, console dump, or another product.
  - Dump title can give clues about the health of CICS or error information that will be available.
- **Status System (ST SYS)**
  - Displays the time the dump was requested, and the difference between Local time and GMT:

```
TIME OF DAY CLOCK: C70F090C C054DA04  12/20/2010 23:57:24.279629 local
TIME OF DAY CLOCK: C70F5984 1DD4DA04  12/21/2010 05:57:24.279629 GMT
Program Producing Dump: SVCDUMP
Program Requesting Dump: DFHKETCB

Incident token: P1       LSYS     12/21/2010 05:57:23.384554 GMT
```

  - Some CICS domains display Local time and others GMT.

What address spaces are included?

- **Cbf rtct**
  - Displays the ASIDs that are included in the dump:

```
SDAS   SDF4   SDF5
       ----   ----   ----
 001   0144   80     00
 002   016C   80     00
```

- **Select all** formats all jobs and assigned address spaces.
- **Select asid(x'144',x'16C')** displays selected ASIDs and the associated jobnames:

```
ASID JOBNAME  ASCBADDR  SELECTION CRITERIA
---- -------- --------  ------------------
0144 CICSAOR  00EE8B00  ASID
016C CICSTOR  00F94880  ASID
```

- When multiple address spaces are dumped, can limit output to just one job with the job=parm:
  - **verbx dfhpd660 'ds=1,job=jobname'**

Browsing the dump

- Option 1 on the IPCS Menu Panel.
- =1 from the IPCS command menu.
- Addresses that begin with an alphabetic character must be preceded by a 0 or followed by a period(.)
- Indirect addressing.
  - **?** - Displays storage at a selected address:

```
ASID(X'010B') ADDRESS(1BE02100.) STORAGE ----------------------------------
1BE02100 ? 1BE02200   00000000   00000000   0004D948   | .\...........R. |
1BE02110   1C93D840   00000000   00000000   00000000   | .lQ ........... |
1BE02120   0000054C   00000000   00000000   80D0357C   | ...<.........}.@ |
```

Entering the ? as above will display the storage at 1BE02200:

```
ASID(X'010B') ADDRESS(1BE02200.) STORAGE ----------------------------------
1BE02200.:1BE0220F. LENGTH(X'10')--All bytes contain X'00'
1BE02210   0000054C   283452F0   0000008A   00000000   | ...<...0........ |
```

CICS Verbexit options through CICS TS 4.1

Keyword		Functional area	
AI	= 0|2	Autoinstall Model Manager	
AP	= 0|1|2|3	Application Domain **APS=<TASKID=nnnnn>**	
AU	= 0|2	CICS affinities utility	
BA	= 0|1|2|3	Business application manager	
BR	= 0|1|2|3	3270 bridge	
CC	= 0|2	CICS catalog domain	
CP	= 0|2	Common Programming Interface	
CQ	= 0|1|2	Auto install model manager	
CSA	= 0|2	CICS Common System Area	
DB2	= 0|1|2|3	The CICS DB2 interface	
DD	= 0|1|2|3	Directory Domain	
DH	= 0|1|2|3	Document handling domain	
DLI	= 0|2	CICS DL/I Interface	
DM	= 0|1|2|3	Domain Manager	
DP	= 0|1|2|3	Debug Profiles manager	(630)
DS	= 0|1|2|3	Dispatcher Domain	
DU	= 0|2	Dump Domain	
EC	= 0|1|2|3	Event Capture domain	(660)
EJ	= 0|1	Enterprise JAVA	
EM	= 0|1|2|3	Event manager domain for BTS	
EP	= 0|1|2|3	Event Processing domain	(660)
FCP	= 0|2	File Control Program	
FT	= 0|1|2|3	CICS WEB Interface	
ICP	= 0|2	Interval Control Program	
IE	= 0|1|2|3	IP ECI Domain	
II	= 0|1|2|3	IIOP	
IND	= 0|1|2|3	Page number indexes for output	
IS	= 0|1|2|3	IP Interconnectivity domain	(650)
JCP	= 0|2	Journal Control Program	
KE	= 0|1|2|3	CICS Kernel	
LD	= 0|1|2|3	Loader Domain	
LG	= 0|1|2|3	Logger Domain	
LM	= 0|1|2|3	Lock Manager domain	
ME	= 0|2	Message domain	
ML	= 0|1|2|3	Markup Language domain	(660)
MN	= 0|1|2|3	Monitoring domain	
MQ	= 0|1|2|3	CICS-MQ interface	(650)

Keyword		Functional area	
MRO	= 0|2	CICS Multi-Region Operation	
NQ	= 0|1|2|3	Enqueue Manager	
OT	= 0|1|2|3	Object Transaction Domain	
PA	= 0|2	Parameter manager domain	
PCP	= 0|2	Program Control Program	
PCT	= 0|2	Program Control Table	
PG	= 0|1|2|3	Program Manager Domain	
PI	= 0|1|2|3	Pipeline Domain	(640)
PR	= 0|2	Partner Resource management	
PT	= 0|1|2|3	Partner Domain	
RD	= 0|2	Resource definition manager	
RL	= 0|1|2|3	Resource Lifecycle domain	(660)
RM	= 0|2	Recovery Management	
RS	= 0|1|2|3	Region Status domain	(660)
RX	= 0|1|2|3	Recoverable EXCI domain	
RZ	= 0|1|2|3	Request Streams	
SH	= 0|1	Scheduler services domain for BTS	
SJ	= 0|1|2|3	JVM Domain	
SM	= 0|1|2|3	Storage Manager domain	
SO	= 0|1|2|3	Sockets domain	
SSA	= 0|2	Static Storage Areas	
ST	= 0|1|2|3	Statistics domain	
SZ	= 0|1	Front End Programming Interface	
TCP	= 0|1|2|3	Terminal Control Program	
TDP	= 0|1|2|3	Transient Data Program	
TI	= 0|1|2|3	Timer domain	
TMP	= 0|2	Table Manager Program	
TR	= 0|1|2|3	Trace domain **TRS=<trace selection parameters>**	
TS	= 0|1|2|3	Temporary Storage Program	
UEH	= 0|2	User Exit Handler	
US	= 0|1|2|3	User Domain	
WB	= 0|1|2	The web interface	
W2	= 0|1|2|3	Web 2.0 domain	(660)
XM	= 0|1|2|3	The transaction manager	
XRF	= 0|2	The extended recovery facility	
XS	= 0|1	Security Domain	

1 – Summary only
2 – Full control block formatting
3 – Both 1 and 2. If you omit the level number, it defaults to level 3 for components that have a summary, and level 2 for those that do not.

CICS Verbexit

- **Verbx dfhpdxxx** where ‘xxx’ is the CICS release:
  - 640 - CICS TS 3.1
  - 650 - CICS TS 3.2
  - 660 - CICS TS 4.1
- When formatting information from a CICS domain or component, there are normally three levels of detail available:
  - 1 – Summary only
  - 2 – Full control block formatting
  - 3 – Both Summary and control blocks
- If the level number is omitted, formatting defaults to level 3 (both summary and control blocks) for components that have a summary, and level 2 for those that don’t.

CSA Time of Day clock (CSA)

- The CICS CSA has a timestamp that is updated each time a task is dispatched on the QR TCB.
  - It is also updated when an application program issues an EXEC CICS ASKTIME request.
- This timestamp is in the form HHMMSSTF Local time, and can be found in the **CSA +x'50'.**
- If CICS is hung, compare the CSA TOD clock to the time of the dump to see when a task was last dispatched on the QR TCB.
- **Verbx dfhpd660 'csa'**

```
CSA 0004E200 Common System Area

0000  00000200 00050020 0004E3A0 86F7EA74  8AE7013C 8004E200 1C276598 1C278158
0020  8B1DDB28 8B1DD6B8 1C2784E0 1C861CF0  0004D948 7F228020 0B1DEB28 1C88B7F0
0040  00052020 1BE08700 2094125C 009A7000  2231233F 1CC19108 00000100 00000000
```

- This corresponds to 22:31:23.3 Local time.

Storage Manager (SM)

- Storage manager summaries show whether CICS is short-on-storage (SOS), above/below the line, above bar.
  - Also shows whether Storage Protection, Reentrant Program Protection, and Transaction Isolation are active.
- **Verbx dfhpd660 'sm=1'**

```
===SM: STORAGE MANAGER DOMAIN - SUMMARY
   SM Domain status:                  INITIALISED
   Storage recovery:                  YES
   Storage protection requested:      YES
   Storage protection active:         YES
   Reentrant program option:          PROTECT
   Transaction isolation requested:   NO
   Transaction isolation active:      NO

   Current DSA limit:                    5120K
   Current DSA total:                    5120K
   Currently SOS below 16M:               YES
   Current EDSA limit:                    500M
   Current EDSA total:                    103M
   Currently SOS above 16M:                NO

   Current GDSA limit:                   2048M
   Current GDSA total:                      4M
   Currently SOS above 2G:                 NO
```

Transaction Manager (XM)

- Transaction manager summaries show:
  - Whether CICS is at MAXTASK:
  - Whether tasks are held for TRANCLASS reasons.
    - Tasks that have been attached but queued for MAXTASK or TCLASS reasons will not be in the CICS Dispatcher or Kernel summaries.
- Transaction manager summaries and control blocks can also be used to determine what time a task was attached.
- **Verbx dfhpd660 'xm=1'**

```
==XM: GLOBAL STATE SUMMARY
   XM domain status:                          Initialised
   Maximum user tasks (MXT):                  90
   System currently at MXT:                   Yes
   XXMATT user exit currently:                Inactive
   XM state lock currently held:              No
   XM trandef state lock currently held:      No
   System attaches delayed for SOS:           No
```

Transaction Summary (XM)

- **Verbx dfhpd660 'xm=1'**

```
==XM: TRANSACTION SUMMARY

Tran Tran  TxnAddr Start Sys  Status   DS       Facility Facility AP
id   num   TxdAddr code  Tran          token    type     token    token
-------------------------------------------------------------------------
CSOL 00003 0E609300 C     Yes  ACT      00040003 None              0E6D6800
           0F659BE0                                               00000000

CEPM 00005 0E609700 C     Yes  ACT      000A0003 None              0E6D7100
           0F60D140                                               00000000
...
BIGD 46840 1BC291B8 T     No   ACT      059020F9 Terminal 1BFA2220 1AF40080
           1BA21920                                               00000000

BIGD 53732 1BC29340 T     No   TCLASS   00000000 None              00000000
           1BA21920               DFHTCL01                       00000000

CWXN 90669 24CA4500 C     No   MXT      00000000 None              00000000
           1912B510                                               00000000

CWXN 90670 24CA4700 C     No   MXT      00000000 None              00000000
           1912B510                                               00000000
```

MXT Summary (XM)

- **Verbx dfhpd660 'xm=1'**
- MXT and TCLASS Summaries are at the end of the formatted summaries.
  - Max to the bottom of the formatted output by entering m followed by PF8.

```
==XM: MXT SUMMARY

   Maximum user tasks (MXT):                90
   System currently at MXT:                 Yes
   Current active user tasks:               90
   Current queued user tasks:               32805
 * Peak active user tasks:                  90
 * Peak queued user tasks:                  32835
 * Times at MXT limit:                      1
```

TCLASS Summary (XM)

- Page backwards from the MXT Summary to see the TCLASS summary:

```
==XM: TCLASS SUMMARY

 Tclass      Max      Purge    Current  Current      Total    Queuing  Queuing      Queuing
  Name     Active   Threshld   Active   Queued    Attaches   TranNum  Transid    Start Time
--------  ------  --------  -------  -------  --------  -------  ----  -----------
DFHTCL01      20         0       20     3707     66475  67732     ORCA  18:37:01.06
                                                       67734     ORCA  18:37:01.13
                                                       67752     ORCA  18:37:02.23
                                                       ...
                                                       90352     ORCA  18:53:14.26
                                                       90356     ORCA  18:53:14.52
                                                       90358     ORCA  18:53:14.65
                                                       90368     ORCA  18:53:15.22
                                                       90370     ORCA  18:53:15.58
                                                       90372     ORCA  18:53:15.63
                                                       90378     ORCA  18:53:16.11
                                                       90382     ORCA  18:53:16.23
                                                       90384     ORCA  18:53:16.25
                                                       90386     ORCA  18:53:16.45
```

Task Attach Time (XM)

- **Verbx dfhpd660 'xm=3'**

```
==XM: TRANSACTION SUMMARY

Tran Tran  TxnAddr Start Sys  Status  DS       Facility Facility AP
id   num   TxdAddr code  Tran         token    type     token    token
-------------------------------------------------------------------
CSOL 00003 1B709300 C     Yes  ACT     00900003 None              1C203800
           1CE97360                                               00000000
```

- FIND on the TXN address. Task attach time is in the **TXN +x'50**':

```
TXN 1B709300 Transaction

0000  01C06EC4 C6C8E7D4 E3A79540 40404040  0101FE08 00000001 00000000 00000000
0020  00000000 00000000 00000000 00000000  00000000 00000000 1B709300 0000003C
0040  1B709700 1B704034 C2E2D6D3 08000000  C56197A6 49D40706 00000000 00000000
```

- **lp ltod C56197A649D40706** will convert the time.

```
01/13/2010 10:25:48.113216 STCK   X'C56197A6 49D40706'
01/13/2010 10:25:48.113216 UTC    X'C56197A6 49D40706'
01/13/2010 04:25:48.113216 LOCAL  X'C561472E EC540706'
```

- Task was attached at 04:25:48.113216 Local time.

Dispatcher (DS)

- Dispatcher summary displays:
  - Tasks currently in the system.
  - Each task's current status - running, suspended, dispatchable , etc.
  - What resources tasks are waiting for.
    - The resource names are described in the CICS InfoCenter (or CICS Problem Determination Guide).
  - What time a task suspended, and whether there is a time-out due.
- Dispatcher summaries and control blocks can be used to determine:
  - How long tasks have been running on their respective TCBs.
  - How long tasks have been dispatchable, waiting for a turn to run on a particular TCB.
- Note: Times in the Dispatcher Summary have not been adjusted for time zone or leap seconds.

Dispatcher Task Summary (DS)

- **Verbx dfhpd660 'ds=1'**

```
==DS: TASKS SUMMARY

 KEY FOR SUMMARY

 T  = TYPE OF TASK           S=SYSTEM  N=NON-SYSTEM
 S  = STATE OF TASK          D=DISPATCHABLE  S=SUSPENDED  R=RUNNING  A=RUNNING ABTERM YES  J=RUNNING IN JVM  E=RESUMED EARLY
 F  = PURGEABILITY FLAG      P=PURGEABLE  N=NOT PURGEABLE
 P  = PURGE STATUS           N=NO PURGE  X=PURGED  P=PURGE PENDING  A=ABTERM PENDING
 TT = TIMEOUT TYPE           IN=INTERVAL  DD=DEADLOCK DELAYED  DI=DEADLOCK IMMEDIATE
 W  = WAIT/SUSPEND TYPE      M=WAIT_MVS  S=SUSPEND  C=WAIT_OLDC  W=WAIT_OLDW
 DTA= DISPATCHER TASK AREA
 AD = ATTACHING DOMAIN
 M  = TASK MODE
```

Shows whether a task is running, suspended, dispatchable etc

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	XM_TXN_TOKEN
00000001	1AAADA98	S	S	N	N	-	ENF	NOTIFY	M	06:16:26.706	-	1B6A3080	DM	1B772C00	RC	1B772C18	
00060003	1AACBA98	S	R									1B6A3500	AP	80051300	CQ		
000A0003	1CECE100	S	S	N	N	-	EPECQEMT	EPSUSPND	M	06:18:17.081	-	1B6A3800	XM	1B709700	EP	1CEC5050	1B7097000**00005**C
000E0003	1B790700	S	S	N	N	-	KCCOMPAT	SINGLE	W	06:19:28.426	-	1B6A3B00	XM	1B70AB00	QR	000418B8	1B70AB000000036C
00820003	1B68D530	S	S	N	N	IN	LGHARTBT	LG_MGRST	S	21:20:37.145	21:20:47.14	1B6C8200		D3C7C8C2	QR	1B6C8200	
00840005	1CFDB100	S	S	N	N	-	CSNC	MROQUEUE	M	21:20:41.828	-	1B6C8380	XM	1B709100	QR	1B79D03C	1B709100000025C
00900003	1B6EDA98	S	S	N	N	-	SODOMAIN	SO_NOWORK	M	21:05:34.378	-	1B6C8C80	XM	1B709300	SL	1B777730	1B709300000003C
00920003	1CF75700	S	S	N	N	-	TCP_NORM	DFHZDSP	W	21:20:39.590	-	1B6C8E00	XM	1B70A100	QR	00056FD0	1B70A100000010C
...																	
0B0209ED	16D9B100	N	S	P	N	-	IRLINK	AOR2S006	M	13:08:56.110	-	20B4B200	XM	18CF2100	QR	7F46B200	18CF2100045441C
07046CFB	1E584900	N	R									1D5F4380	XM	1EDA6100	QR		1EDA6100043738C
07065887	1E4BA100	N	S	N	N	-		LMQUEUE	S	20:00:30.503	-	1D5F4500	XM	2170DD00	QR	1D395200	2170DD000440047C
070C338D	1E48E100	N	D									1D5F4980	XM	2170DB00	L8		2170DB000440046C
07040001	2BDC1480	N	S	P	N	-	**ENQUEUE**	**FCDSRECD**	S	**20:01:11.363**	-	2B0E2380	XM	2E40C030	QR	2B0E2380	2E40C030043203C
0782000B	1E5DE900	N	S	N	N	-	EKCWAIT	SINGLE	W	05:08:26.187	-	1D5F6200	XM	1D609B00	QR	00006058	1D609B00000076C
07860005	1E5FE900	N	S	P	N	-			M	18:24:57.263	-	1D5F6500	XM	1D60AB00	QR	1E61E33C	1D60AB00000054C
0A9202AD	16E23500	N	S	P	N	-	IRLINK	AOR2S003	M	13:08:25.177	-	20B4AE00	XM	18EF1900	QR	7F46B320	18EF1900044796C
...																	

Task #

Shows what TCB a task is running on, suspended on, or waiting to run on

What time did a task become ‘Running’ or Dispatchable?

- Dispatcher Task Summary shows the time a task was suspended, but not what time a task became ‘Running’ or ‘Dispatchable’.
  - Tasks that are ready to run but the dispatcher hasn’t dispatched them yet are ‘Dispatchable’.
- Dispatcher Task Summary includes the address of the Dispatcher Task Area (DTA) control block.
  - DTA has two clock fields at +x’50’ and +x’58’.
- When a task is ‘Dispatchable’, the **DTA +X’50’** contains the time the task became dispatchable.
  - For a suspended task, DTA +X’50’ is also the time the task was suspended.
- For a running task, **DTA +X’58’** is within .1 second of the time the task became ‘running’ (i.e. was dispatched).

What time did a task become Running ? (DS)

- **Verbx dfhpd660 'ds=3'**

```
DS_TOKEN KE_TASK  T S F P TT RESOURCE RESOURCE   W TIME OF      TIMEOUT     DTA      AD ATTACHER M SUSPAREA XM_TXN_TOKEN
                            TYPE     NAME       SUSPEND      DUE         (DSTSK)     TOKEN
...
07046CFB 1E584900 N R                                                    1D5F4380 XM 1EDA6100 QR          1EDA6100043738C
07065887 1E4BA100 N S N N -          LMQUEUE    S 20:00:30.503      -     1D5F4500 XM 2170DD00 QR 1D395200 2170DD00044047C
070C338D 1E48E100 N D                                                    1D5F4980 XM 2170DB00 QR          2170DB00044046C
020A0003 20FCE100 S D                                                    30288800 XM 20FCC900 QR          20FCC900043202C
020C0001 20FCE500 S D                                                    30288980 XM 20FCCB00 QR          20FCCB00043276C
020E0001 20FCE900 S D                                                    30288B00 XM 20FCCD00 QR          20FCCD00043373C
02100003 20FD2100 S D                                                    30288C80 XM 20FCC500 QR          20FCC500043444C
0A9202AD 16E23500 N S P N -  IRLINK   AOR2S003   M 13:08:25.177      -     20B4AE00 XM 18EF1900 QR 7F46B320 18EF1900044796C
```

- **FIND on the DTA address to get to the formatted DTA control block. Check the DTA +x'58':**

```
DTA 1D5F4380 DISPATCHER TASK AREA

0000  FFFFFFFF 07046CFB 40404040 40404040  D3D4D8E4 C5E4C540 40404040 40404040 *......%.        LMQUEUE         *
0020  1D5F4380 00040000 FE000000 1D5AD080  FFFFFFFF 00000000 FFFFFFFF FFFFFFFF *.^...........!}..................*
0040  00000000 02FF0000 DFFFFFFF 1DD12A53  C6C67656 4B8F5F84 C6C67656 4F3AE284 *.............J..FF....^dFF..|.Sd*
```

- **lp ltod C6C676564F3AE284 will convert the time to show that the running task became 'Running' within .1 sec of:**

```
10/24/2010 06:37:39.073966 STCK   X'C6C67656 4F3AE284'
10/24/2010 06:37:39.073966 UTC    X'C6C67656 4F3AE284'
10/24/2010 01:37:39.073966 LOCAL  X'C6C63348 2BFAE284'
```

What time did a task become Dispatchable ?

- **Verbx dfhpd660 'ds=3'**

```
DS_TOKEN KE_TASK  T S F P TT RESOURCE RESOURCE   W TIME OF      TIMEOUT     DTA      AD ATTACHER M SUSPAREA XM_TXN_TOKEN
                            TYPE     NAME       SUSPEND       DUE         (DSTSK)     TOKEN
...
07046CFB 1E584900 N R                                                    1D5F4380 XM 1EDA6100 QR          1EDA6100043738C
07065887 1E4BA100 N S N N -          LMQUEUE    S 20:00:30.503      -     1D5F4500 XM 2170DD00 QR 1D395200 2170DD00044047C
070C338D 1E48E100 N D                                                    1D5F4980 XM 2170DB00 QR          2170DB00044046C
020A0003 20FCE100 S D                                                    30288800 XM 20FCC900 QR          20FCC900043202C
020C0001 20FCE500 S D                                                    30288980 XM 20FCCB00 QR          20FCCB00043276C
020E0001 20FCE900 S D                                                    30288B00 XM 20FCCD00 QR          20FCCD00043373C
02100003 20FD2100 S D                                                    30288C80 XM 20FCC500 QR          20FCC500043444C
0A9202AD 16E23500 N S P N -  IRLINK   AOR2S003   M 13:08:25.177      -     20B4AE00 XM 18EF1900 QR 7F46B320 18EF1900044796C
```

- **FIND on the DTA address to get to the formatted DTA control block. Check the DTA +x'50':**

```
DTA 1D5F4980 DISPATCHER TASK AREA

0000  FFFFFFFF 070C338D C4E2E3E2 D2C4C5C6  070C338D E3E9F140 40404040 40404040 *........DSTSKDEF....TZ1        *
0020  1D5F4980 00010000 FE000000 1D5FF680  1D5F3200 00000000 FFFFFFFF FFFFFFFF *.^............^6..^..............*
0040  00000000 03FF0000 3938922F 9F7B81FB  C6C66DEF 925B7E05 C6C5C0CF 5E82D285 *..........k..#a.FF_.k$=.FE{.;bKe*
```

- **IP LTOD C6C66DEF925B7E05 will convert the time to show that the task became 'Dispatchable' at:**

```
10/24/2010 06:00:03.861943 STCK   X'C6C66DEF 925B7E05'
10/24/2010 06:00:03.861943 UTC    X'C6C66DEF 925B7E05'
10/24/2010 01:00:03.861943 LOCAL  X'C6C62AE1 6F1B7E05'
```

Kernel (KE)

- **Verbx dfhpd660 'ke=1'**
- Task Summary
  - Includes all the tasks currently in the system.
  - Also includes a default task for each MVS TCB managed by the Kernel.
  - Shows tasks that are ***Running** if the task is currently dispatched on a TCB.
  - Shows tasks flagged with an error (when any kernel stack for the task has been flagged with an error).
- Kernel Error Data (if applicable)
- Kernel Stacks
- Kernel Error Table Summary
  - Formats the 50 most recent errors in CICS.

Kernel Task Summary (KE)

- **Verbx dfhpd660 'ke=1'**

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

KE_NUM KE_TASK  STATUS       TCA_ADDR TRAN_# TRANSID DS_TASK  KE_KTCB  ERROR

0001   0E38FA98 KTCB Step    00000000                00000000 0E3D3FE0
0002   0E38F530 KTCB QR      00000000                0E603100 0E3D6FF8
0003   0E39EA98 KTCB RO      00000000                0E603200 0E3D5FF0
0004   0E39E530 KTCB FO      00000000                0E603300 0E3D4FE8
0005   0E3ADA98 Not Running  00000000                0E582080 0E3D5FF0
0006   0E3AD530 Not Running  0E6DC100 00040  CSNE    0E582200 0E3D6FF8
0007   0E3BCA98 KTCB SL      00000000                0E603500 0E5BEFF8
0008   0E3CBA98 ***Running** 00000000                0E582680 0E58CFF8
0009   0E3CB530 KTCB EP      00000000                0E603700 1220EFF8
000A   0F7A7100 Not Running  0E6DB800 00038  CISE    0E582980 0E3D6FF8
000B   0F724100 Not Running  0E6DC800 00039  CISM    0E5A7B00 0E3D6FF8
...
002A   12274A98 KTCB CQ      00000000                0E603400 0E58CFF8
...
0042   0F4EE700 ***Running** 00062700 00069  BIGD    0E5F5080 0E3D6FF8 *YES*
0043   0F4FD700 Not Running  0E6DA800 00061  ORCA    0E5F5800 0E3D6FF8
0044   0F4FE700 Not Running  00061080 00047  BIGE    0E5F5500 0E3D6FF8
0045   0F4FF100 Not Running  0005F080 00046  CEMT    0E5F5380 0E3D6FF8
```

- Key fields are the Kernel number, transaction number, transaction ID; if task flagged as ***Running** or in error.
- Field KE_KTCB identifies the TCB a task is running on.

Kernel Error Table Summary (KE)

- **Verbx dfhpd660 'ke=1'**
  - Max to the bottom to see the Kernel Error Table summary.

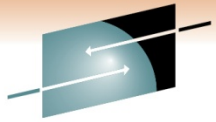
```
==KE: KE Domain Error Table Summary

ERR_NUM    ERR_TIME   KE_NUM   ERROR TYPE              ERR_CODE   MODULE    OFFSET
=======    ========   ======   ==========              ========   ======    ======

00000001   10:56:35    0059    TRAN_ABEND_PERCOLATE    ---/AFCY   DFHPCP    000005E8
00000002   10:56:35    0059    TRAN_ABEND_PERCOLATE    ---/AFCY   DFHEIFC   00001298
00000003   10:56:35    0059    TRAN_ABEND_PERCOLATE    ---/AFCY   DFHEIP    000008AC
00000004   11:32:40    0010    PROGRAM_CHECK           0C1/AKEA   UNKNOWN   UNKNOWN
00000005   11:32:40    0010    TRAN_ABEND_PERCOLATE    ---/ASRA   DFHSR1    00000598
00000006   11:41:34    0024    TRAN_ABEND_PERCOLATE    ---/ATNI   DFHPCP    000005E8
00000007   11:41:34    0024    TRAN_ABEND_PERCOLATE    ---/ATNI   DFHEIP    000024E8
00000008   11:41:34    0024    TRAN_ABEND_PERCOLATE    ---/ATNI   DFHEPC    00000246
00000009   11:42:58    0010    PROGRAM_CHECK           0C4/AKEA   UNKNOWN   UNKNOWN
```

- Kernel Error Table holds a maximum of the 50 most recent errors.
- ERR_CODE is the system abend code, if any, followed by the user abend code. (CICS abend codes begin with the letter 'A').
- Note the KE_NUM to identify errors associated with a particular kernel task.
  - May need to check task attach times to determine if errors associated with a kernel task are for the same task or different tasks.

Kernel Error Data (KERRD)

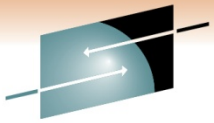
- **Verbx dfhpd660 'ke=1'**
- If the dump is taken for an error, Kernel Error Data will be formatted with the summary information.
  - If not, the most recent 50 errors can be found with:
  - **verbx dfhpd660 'ke=2'**
  - Find the error using the ERR_NUM identified in the Error Table Summary. e.g. **FIND 'number: 00000009'**
    - *Note: 2 spaces between the colon and the error number.*
- The dump formatter provides the raw KERRD control block which contains PSW, registers, and error information.
- The dump formatter also formats key areas:
  - PSW and registers, together with storage in the vicinity of the PSW and register addresses.
  - The module in control at time of error, from a CICS perspective.
    - If the error did not occur in the controlling program, the offset shown will be negative. i.e. FFFFFFFF
  - The execution key and space (Basespace or Subspace if Transaction Isolation is active).

Kernel Error Data (KERRD)



- There are two sets of PSW and Registers, CICS and MVS.
- These two save areas are different only when CICS has issued an SVC (service request), and an abend or program check occurs during execution of the code processing the SVC.
  - CICS PSW and registers will reflect the code that issued the SVC.
  - MVS PSW and registers will reflect the actual program check or abend.
    - MVS PSW and registers will only be formatted if they are different than the CICS PSW and registers.
  - The exception is when CICS has detected a runaway (looping) task. CICS issues an abend 999 to terminate the task:
    - CICS PSW and registers will point within the looping module.
    - MVS PSW and registers will point within Kernel module DFHKETIX after the abend SVC.
- BEAR – Breaking Event Address Register, is also formatted.
  - BEAR contains the address of the last branch instruction taken prior to a program check or abend. (Useful for wild branch detection).
  - Formatted as **Branch Event Address**.

Example Kernel Error Data (KERRD)




```
=KE: Error Number:  00000009
KERRD 1A3D69D0 KERNEL ERROR DATA

0000  F0C3F461 C1D2C5C1 018400C4 0000FFFF  C4C6C8D9 D4C4D440 18D61FA8 193D0800 *0C4/AKEA.d.D..DFHRMDM .O.y....*
0020  0005F080 1A3D6700 00000001 00000010  078D0000 B92F8A7A 00040010 3D711000 *..0.................:........*
0040  B92F8A7A 80800000 00000000 18E1EF30  00000000 1A3E5468 00000000 00007000 *...:.........................*
0060  00000000 98D61FA8 00000000 00076734  00000000 98CF7E20 00000000 388E2400 *....qO.y.........q.=.........*
0080  00000000 B92F87F0 00000000 04418558  00000000 B92F97F0 00000000 1A3E5468 *......g0....e.......p0........*
00A0  00000000 0000B8F0 00000000 0005F080  00000000 1A3E4F90 00000000 98D64AA2 *.......0....0.......|.....qO¢s*
00C0  00000000 B92F8940 983315D6 00000000  00000000 00000000 00000000 00000000 *......i q..O...................*
00E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000 *................................*
0100  00000000 00000000 078D0000 B92F8A7A  00040010 3D711000 B92F8A7A 80800000 *.............:...........:....*
0120  00000000 18E1EF30 00000000 1A3E5468  00000000 00007000 00000000 98D61FA8 *.........................qO.y*
0140  00000000 00076734 00000000 98CF7E20  00000000 388E2400 00000000 B92F87F0 *.........q.=.............g0*
0160  00000000 04418558 00000000 B92F97F0  00000000 1A3E5468 00000000 0000B8F0 *......e.....p0..............0*
0180  00000000 0005F080 00000000 1A3E4F90  00000000 98D64AA2 00000000 B92F8940 *......0.....|.....qO¢s.....i *

 Error Code:  0C4/AKEA    Error Type:  PROGRAM_CHECK          Timestamp:  C6BC0F99D77ACF04
Date (GMT)   :  16/10/10        Time (GMT)   :  00:04:46.267308
Date (LOCAL) :  15/10/10        Time (LOCAL) :  18:04:46.267308

KE_NUM:  0010    KE_TASK:  0F4EE700    TCA_ADDR:  00062700    DS_TASK:  0E5F5080

Program DFHRMDM was in control, but the PSW was elsewhere.
Error happened under the CICS RB.
CICS Registers and PSW.

PSW:  078D0000 B92F8A7A   Instruction Length:  4  Interrupt Code:  10  Exception Address:  3D711000

Execution key at Program Check/Abend: 8
Space at Program Check/Abend: Basespace          Branch Event Address: 00000000_392F8950

 64-BIT REGISTERS 0-15
 0000  00000000 18E1EF30 00000000 1A3E5468  00000000 00007000 00000000 98D61FA8
 0020  00000000 00076734 00000000 98CF7E20  00000000 388E2400 00000000 B92F87F0
 0040  00000000 04418558 00000000 B92F97F0  00000000 1A3E5468 00000000 0000B8F0
 0060  00000000 0005F080 00000000 1A3E4F90  00000000 98D64AA2 00000000 B92F8940
```

Breaking Event Address Register (BEAR)

- Example using BEAR to diagnose a wild branch:

```
KERRD 1A3D69D0 KERNEL ERROR DATA

  Error Code:  0C1/AKEA    Error Type:  PROGRAM_CHECK

Program DFHAPLI1 was in control, but the PSW was elsewhere.

CICS Registers and PSW.

PSW: 078D0000 00000002   Instruction Length: 2  Interrupt Code: 01  Exception Address: 000A0000

 Execution key at Program Check/Abend: 8

 Space at Program Check/Abend: Basespace          Branch Event Address: 00084A88

  REGISTERS 0-15

  0000  00000000 7FFFF000 7FFFF000 7FFFF000  7FFFF000 7FFFF000 7FFFF000 00000000
  0020  00000000 7FFFF000 7FFFF000 00000000  00000000 00000000 00000000 00000000


   00083250   00000000  00000000  5CC4C6C8  C5C9D740  | ........*DFHEIP  |
   00083260   40000833  70F0F6F4  F0C91154  1011E4D2  |  ....0650I....UK |
   00083270   F3F0F1F3  F3401400  C3C9C3E2  40F5F6F5  | 52360 ..CICS 565 |
   00083280   F560D4F1  F5404DC3  5D40C3D6  D7E8D9C9  | 5-M15 (C) COPYRI |
   ...
   00084A80   B20A3000  98ECD00C  0B0E58E0  C00C58EE  | ....q.}....\{... |
```

Bear points to this Branch instruction.
Prior to the Branch is an instruction loading the above registers.

Kernel Stacks (KE)

- Kernel stacks show the sequence of CICS modules currently involved with a task.
  - Provides a perspective of what the task is doing in CICS.
- Kernels stacks contain register save areas and working storage for each CICS module.

```
KE_NUM @STACK   LEN  TYPE ADDRESS  LINK REG OFFSET ERR NAME
00B9   1EB5C020 0120 Bot  9D001C00 9D001FBC 0003BC     DFHKETA
00B9   1EB5C140 0320 Dom  9D01A660 9D01A878 000218     DFHDSKE
00B9   1EB5C460 0820 Dom  9D042C58 9D043E24 0011CC     DFHXMTA
00B9   1EB5CC80 05D0 Dom  9DB0B9C0 9DB0C97A 000FBA     DFHPGPG
                     Int   +0002DC 9DB0BB52 000192     INITIAL_LINK
00B9   1EB5D250 0AD0 Dom  9DD0C700 800837A4 000000     DFHAPLI1
                     Int   +002FD2 9DD0D1A4 000AA4     LE370_INTERFACE
                     Int   +002DAC 9DD10764 004064     INVOKE_FOR_RECURSION
00B9   1EB5DD20 04A0 Sub  9DCE5000 9DCE6D66 001D66     DFHEIFC
                     Int   +00141A 9DCE54D8 0004D8     CALL_FCFR
00B9   1EB5E1C0 0860 Dom  9E162200 9E16592A 00372A     DFHFCFR
                     Int   +00B31A 9E163AE0 0018E0     IMPLICIT_OPEN
                     Int   +0036EA 9E16D94A 00B74A     OPEN_FILE
00B9   0005F020 0A50 Sub  9E244200 9E24AF96 006D96     DFHFCFS
                     Int   +00575E 9E245336 001136     OPEN_FILE
                     Int   +006D60 9E249A82 005882     SETOPEN
00B9   00060020 0ED0 Sub  9E294100 9E29AE4A 006D4A     DFHFCRO
                     Int   +000382 9E29434C 00024C     OPEN_FILE
                     Int   +004554 9E294AB0 0009B0     PERFORM_VSAM_OPEN
```

Finding the MVS TCB address (KE)

- Often we need to find the MVS TCB for one of the CICS TCBs:
- **Verbx dfhpd660 'ke=3'**
- Use the selected KE_KTCB from the Kernel Task Summary.
- FIND the formatted KTCB control blocks.
- For the selected KTCB address, **KTCB+x'50'** contains the address of the MVS TCB:

```
KTCB 0E3D5FF0 KTCB TABLE ENTRY

0000  D2E3C3C2 40404040 00000000 0E39EA98  0E39EA98 0E39F020 00000000 5166F0AE *KTCB    ..q...q..0.......0.*
0020  00000000 7D000000 00000000 00000000  80000001 00000000 06820000 D900D9D6 *....'................b..R.RO*
0040  00000000 00000000 0E3D5FF0 40000000  008E5920 00000000 00006130 00000000 *.........^0 ............/.....*
```

- For one of the more common TCBs, like the QR, can also use the trace domain to quickly find the TCB address:
- **Verbx dfhpd660 'tr=2'** then enter **FIND QR**

```
 AP 00E1 EIP EXIT SUSPEND OK        REQ(00F4) FIELD-A(00000000 ....) FIELD-B(00001208 ....)
    TASK-00069 KE_NUM-0042 TCB-QR /008E55F8 RET-9061A2F2 TIME-18:04:46.2613956572
```

Loader domain (LD)

- **Verbx dfhpd660 'ld=1'**
- Program Repertoire provides program length, use count etc.

```
==LD: PROGRAM REPERTOIRE

PGM NAME USE CNT USERS LOADS COPIES LENGTH  USE TYP ATTRB EXEC R/A MODE   DEFINITION       CPE      PLIBE  STATUS
                                                      KEY  OVERRIDE  DATE     TIME    ADDRESS  ADDRESS

DFHERM   1       1     1     1     00005708 NUC ANY RSDNT CICS  -    -  3/08/10 21:29:57 0E6AFAC0 0E652030 LOADED
DFHERMSP 2       2     1     1     00001600 NUC ANY RSDNT CICS  -   31  3/08/10 21:29:58 0E6AFC60 0E652030 LOADED
DFHESC   1       1     1     1     00000668 NUC ANY RSDNT CICS  -    -  3/08/10 21:29:57 0E6AFD30 0E652030 LOADED
...
PROGRAM1 0       0     0     0     00000D20 APP RPL REUSE USER  -    -  9/09/10 10:45:01 0F5F2370 0E652030 LOCATED
PROGRAM2 0       0     0     0     000019F0 APP RPL REUSE USER  -    -  9/09/10 10:45:01 0F5F2440 0E652030 LOCATED
PROGRAM3 0       0     0     0     00002D50 APP RPL REUSE USER  -    -  9/09/10 10:45:01 0F5F2510 0E652030 LOCATED
```

- Program Storage Map provides a program's load point, entry point, PTF level (if applicable).

```
==LD: PROGRAM STORAGE MAP

PGM NAME ENTRY PT  CSECT    LOAD PT. REL. PTF LVL LAST COMPILED  COPY NO USERS LOCN TYP ATTRIBUTE R/A MODE APE ADDR

DFHAPXM  8008EF00 DFHAPXM  0008EF00 660  HCI6600 05/30/09 02.29    1      2   RDSA ANY RESIDENT   -   31  0E61C030
                  IPRAPXM  00090060
DFHDLI   80090314 DFHDLI   00090200 0660 HCI6600 I  30/05 03.06    1      1   RDSA ANY RESIDENT   -    -  0E61C1F8
                  IPRDLI   000913E8
DFHEITHG 00091620 -noheda- 00091600                                 1      1   RDSA ANY RESIDENT   -    -  0E61E030
DFHERM   00098914 DFHERM   00098800 0660 UK48726 I  28/07 11.10    1      1   RDSA ANY RESIDENT   -    -  0E61F880
                  DFHERMS  0009CE28 0660 UK48726 I  28/07 11.10
                  DFHTIEM  0009D418 0660 HCI6600 I  30/05 05.09
```

Program domain (PG)

- The Program domain has a Program Level control block (PLCB) that shows the sequence of application programs, Task Related User Exits (TRUEs) and Global User Exits (GLUEs) that are involved with a task.
- **Verbx dfhpd660 'pg=1'**
- FIND on the task number:

```
==PG: PTA SUMMARY FOR TRAN NUM : 44444,  PTA ADDRESS : 153F56F0
LOG-LVL : 3         SYS-LVL : 0         TASK-LLE : 1655F160  PLCB : 16029268
=PG: TASK LLE SUMMARY
LLE-ADDR    PROGRAM     PPTE-ADD
1655F160    COBLPGM1    16666D98
1655F1F0    COBLPGM2    16427348
1655F5E0    COBLPGM3    1665ADF0

=PG: TASK PLCB SUMMARY
PROG DFHD2EX1 LVL 3 PLCB 16029268 LD 00000000 ENT 00000000 LEN 000000 PPTE 1622FD98 ENV TRUE INV PROGRAM2 EXIT
 PROGRAM: DFHD2EX1 CPE: 162966B0 LIB: DFHRPL   CONCAT: 00

PROG PROGRAM2 LVL 2 PLCB 16171E68 LD 18AF8F10 ENT 98AF8F10 LEN 003370 PPTE 164A6AD8 ENV EXEC INV PROGRAM1 EXIT
  COMMAREA 163A3878 LEN 77AA STORAGE U
 PROGRAM: PROGRAM2  CPE: 16680C60 LIB: DFHRPL   CONCAT: 0E

PROG PROGRAM1 LVL 1 PLCB 1616F230 LD 18AFD000 ENT 98AFD13C LEN 0022B8 PPTE 16266C90 ENV EXEC INV CICS     EXIT
 PROGRAM: PROGRAM1  CPE: 15383ED0 LIB: DFHRPL   CONCAT: 00
```

Program Level summary (PG)

- The LVL 1 program, PROGRAM1, is the initial program.
  - It's environment shows EXEC indicating a normal application program.
  - PROGRAM1 issues an EXEC CICS LINK to PROGRAM2.
- The LVL 2 program is PROGRAM2.
  - PROGRAM2 issues an SQL call.
- The LVL 3 program is DFHD2EX1, running as a Task Related User Exit (TRUE).
- PLCB summary is also useful for providing program name, length, and load point information.
- Note: Programs listed in the LLE Summary have been loaded on behalf of the current task. e.g. via COBOL call

LMQUEUE – Waiting on a lock

- **Verbx dfhpd660 'ds=1'**
- Dispatcher domain shows that tasks are in LMQUEUE suspends, indicating that tasks are waiting on a lock(s):

```
DS_TOKEN KE_TASK  T S F P TT RESOURCE RESOURCE_NAME W TIME OF      TIMEOUT  DTA      AD ATTACHER M SUSPAREA XM_TXN_TOKEN
                            TYPE                    SUSPEND      DUE      (DSTSK)     TOKEN
...
0512813D 1E59C100 N S N N -           LMQUEUE       S 20:07:00.500    -     1D58FE00 XM 226FF300 QR 1D395470 226FF3000044071C
05806301 1E5FF900 N S N N -           LMQUEUE       S 20:16:02.558    -     1D590080 XM 22755100 QR 1D395620 227551000044092C
0586B227 1E5B2900 N S N N -           LMQUEUE       S 20:58:22.757    -     1D590500 XM 2170D100 QR 1D582920 2170D1000044040C
058E7DA7 1E59C500 N S N N -           LMQUEUE       S 20:37:34.170    -     1D590B00 XM 22751700 QR 1D3958F0 227517000044131C
0600E50B 1E52A100 N S N N -           LMQUEUE       S 20:58:10.436    -     1D5F2080 XM 21713D00 QR 1D582B60 21713D000044039C
0604D7A5 1E56E100 N S N N -           LMQUEUE       S 20:09:01.663    -     1D5F2380 XM 226FFB00 QR 1D395500 226FFB000044076C
060A2A1D 1E48E900 N S N N -           LMQUEUE       S 20:57:59.773    -     1D5F2800 XM 20EB9B00 QR 1D582800 20EB9B000044034C
060C8D99 1E4A4500 N S N N -           LMQUEUE       S 20:05:26.882    -     1D5F2980 XM 226FC700 QR 1D3953B0 226FC7000044061C
060ED8AF 1E540900 N S N N -           LMQUEUE       S 20:18:03.300    -     1D5F2B00 XM 22755B00 QR 1D3956E0 22755B000044101C
06824099 1E59C900 N D                                                       1D5F3200 XM 2173A700 QR          2173A7000044051C
0686199D 1E558900 N S N N -           LMQUEUE       S 20:00:47.058    -     1D5F3500 XM 2173A300 QR 1D395260 2173A3000044049C
07029D95 1E5F6500 N D                                                       1D5F4200 XM 20FCC300 QR          20FCC3000043505C
07046CFB 1E584900 N R                                                       1D5F4380 XM 1EDA6100 QR          1EDA61000043738C
07065887 1E4BA100 N S N N -           LMQUEUE       S 20:00:30.503    -     1D5F4500 XM 2170DD00 QR 1D395200 2170DD000044047C
070C338D 1E48E100 N D                                                       1D5F4980 XM 2170DB00 QR          2170DB000044046C
07040001 15D92500 N S P N - ENQUEUE   FCDSRECD      S 20:01:11.363    -     2B0E2380 XM 2E40C030 QR 2B0E2380 2E40C0300043203C
070EFAFF 1E584100 N S N N -           LMQUEUE       S 20:59:32.626    -     1D5F4B00 XM 2170D900 QR 1D3951A0 2170D9000044044C
0710FCDF 1E4E6100 N S N N -           LMQUEUE       S 20:05:07.555    -     1D5F4C80 XM 226FC300 QR 1D395350 226FC3000044059C
07126B47 1E540100 N S N N -           LMQUEUE       S 20:06:13.407    -     1D5F4E00 XM 226FCB00 QR 1D395410 226FCB000044065C
0782000B 1E5DE900 N S N N - EKCWAIT   SINGLE        W 05:08:26.187    -     1D5F6200 XM 1D609B00 QR 00006058 1D609B000000076C
07840007 1E5FF500 N S P N -                         M 05:08:25.530    -     1D5F6380 XM 1D60A100 QR 1E62C33C 1D60A1000000053C
07860005 1E5FE900 N S P N -                         M 18:24:57.263    -     1D5F6500 XM 1D60AB00 QR 1E61E33C 1D60AB000000054C
07880003 1E5FE500 N S N N -           LMQUEUE       S 20:04:02.187    -     1D5F6680 XM 1E941100 QR 1D395290 1E9411000044055C
...
```

Lock Manager (LM)

- **Verbx dfhpd660 'lm=1'**
- FIND on keyword 'wait' to find the Lock Wait Queue and see what lock(s) tasks are waiting for:

```
==LM: LOCK WAIT QUEUE

LOCK       ADDRESS   -> NEXT   OWNER     MODE  SUSPEND   STATUS
NAME                                             TOKEN
----       -------   -------   -----     ----  -------   -------
DUDATSET   15B991E4  15B99234  16E67100  EXCL  00930003
TSLOCK     1D5BDE04  1D5BDDF0  1E59C500  EXCL  015B0001
           1D5BDDF0  1D5BDDDC  1E4FC500  EXCL  01590001
           1D5BDDDC  1D5BDDC8  1E4E6900  EXCL  01570001
           1D5BDDC8  1D5BDDB4  1E5C8500  EXCL  01550001
           1D5BDDB4  1D5BDDA0  1E5B2100  EXCL  01530001
           1D5BDDA0  1D5BDD8C  1E5B2500  EXCL  01510001
           1D5BDD8C  1D5BDD78  1E4BA900  EXCL  014F0001
           1D5BDD78  1D5BDD64  1E4BA500  EXCL  014D0001
           1D5BDD64  1D5BDD50  1E48E500  EXCL  014B0001
           1D5BDD50  1D5BDD3C  1E5C8900  EXCL  01490001
           1D5BDD3C  1D5BDD28  1E584900  EXCL  01470001
           1D5BDD28  1D5BDD14  1E540900  EXCL  01450001
           ...
```

Lock Manager (LM)

- FIND on the lock name PREV to see the Allocated Locks summary and determine what task currently owns the lock:

```
==LM: ALLOCATED LOCKS

LOCK       LOCK       OWNER      MODE  COUNT      # LOCK     # LOCK     -> QUEUE
NAME       TOKEN                                  REQUESTS   SUSPENDS
----       -----      -----      ----  -----      --------   --------   --------
DUDATSET   15530020   16DDF900   EXCL                   181         80   15B991E4
...
TSLOCK     34603DD0   15D92500   EXCL                327477    3648517   1D5BDE04
```

- The address listed as the lock owner identifies a Kernel TAS control block address.

Identify the Lock Owner

- Format the Kernel summary and FIND on the TAS address to identify the owning task.
- **Verbx dfhpd660 'ke=1'**

```
KE_NUM KE_TASK   STATUS       TCA_ADDR TRAN_# TRANSID DS_TASK   KE_KTCB   ERROR

0038   15D92500 Not Running  150DA700 42303  BIGD    344DB680 35668FF8
```

- Format the Dispatcher summary and find the task (using the TAS address or the Task #), and see what the lock owner is waiting for.
- **Verbx dfhpd660 'ds=1'** (some fields deleted)

```
KE_TASK   T S F P TT RESOURCE RESOURCE_NAME  TIME OF     TIMEOUT  DTA      M SUSPAREA XM_TXN_TOKEN
                    TYPE                    SUSPEND     DUE      (DSTSK)

15D92500 N S P N -  ENQUEUE  FCDSRECD       18:41:51.98    -     22128B00 QR 22128B00 2071E7000042303C
```

Enqueue Domain (NQ)

- If Dispatcher summary shows a task waiting on an ENQUEUE:

```
KE_TASK  T S F P TT RESOURCE RESOURCE_NAME  TIME OF      TIMEOUT   DTA      M SUSPAREA XM_TXN_TOKEN
                    TYPE                    SUSPEND      DUE       (DSTSK)

15D92500 N S P N -  ENQUEUE  FCDSRECD      18:41:51.988      -   22128B00 QR 22128B00 2071E700042303C
```

- **Verbx dfhpd650 'nq=1'**
- FIND on the ENQUEUE name/type. e.g. FCDSRECD

```
==NQ: ENQUEUE POOL SUMMARY - FCDSRECD

   Default shunt action:                    Retain
  *Total enqueue requests:                  2345
  *Total requests that have waited:         65
  *Total requests failed busy:              47
  *Total requests failed locked:            0
  *Total requests timed out:                0
  *Total enqueues that were retained:       0

*NOTE: These values were reset at 04:57:03 (last statistics interval collection)
```

ENQUEUE Summary FCDSRECD (NQ)

- FIND on your task#. Task # 42303 is waiting on a file record lock owned by a KROO task, # 42046:

```
                                                    OWNER  /  WAITER
                                          NQEA     Tran  Tran                          Lifetime  Hash
          Enqueue Name                Len Sta Address    id   Num    Local Uowid      Uow Tsk   Indx
------------------------------------- --- --- -------- ---- ----- ----------------- --- ---   ----

X'10B70440'1000000000000000000  26 Act 0FB7EE40 KROO 42046 BB66EA4300225621   1   0    10
          X'F1F0F0F0F0F0F0F0F0F0F
                                Waiter :    0FB7ED80 CSMI 46047 BB66EA431739D1A1   1   0    10
                                Waiter :    13CA8480 TTFN 42545 C3F038A5953D5204   1   0    47
                                Waiter :    13CA5900 BIGD 42303 C3F038B4CEE9D404   1   0    47

X'10B70440'CCCCCCCCCCCCCCCCCCCC  26 Act 13CA5C00 BIGD 42303 C3F036ABB02D8E06   2   0    47
          X'C3C3C3C3C3C3C3C3C3C3C
                                Waiter :    13CA2D80 TTFN 51124 C3F03700BC9BB046   1   0    47
                                Waiter :    1234D380 ROFL 51793 C3F0371F52060486   1   0    47
```

- For an ENQUEUE type FCDSRECD, the first word of the ENQUEUE name (x'10B70440') is the address of the Dataset Name Block, followed by the key of the record (or a portion of the key. Entire key can be found in the NQEA).

File Control component (FCP)

- **Verbx dfhpd660 'fcp=1'**
- Find on the dataset name block address found in the NQ domain to identify the CICS file associated with the ENQUEUE request..

```
ADDRESS  FILENAME ACC  TYPE  MODE RLS LSR REM SLG SREQS STATUS   JID DSNB-OBJ DSNB-BAS FR
-------------------------------------------------------------------------------------------
10B79B28 BIGG     VSAM KSDS  BASE NO  1   NO  YES RUADB OPEN ENA 0   10B70440 10B70440 YES
```

Application domain (AP)

- The Application domain can be formatted with:
  - **Verbx dfhpd660 'ap=3'**
- All task storage acquired on behalf of a task is formatted.
  - Any storage violations currently detected by CICS are flagged when the leading and trailing storage accounting areas don't match.
- Task-related control blocks including the Task Control Area (TCA), and EXEC CICS request areas are formatted.
  - EXEC Interface Block (EIB) and EXEC Interface User Structure (EIUS) can be used to find the most recent EXEC CICS request for a task, and identify the program that made the request.
- Can selectively format storage associated with a single task:
  - **Verbx dfhpd660 'aps=<taskid=xxxxx>**
  - If available, will also format LE/370 storage information for the task using the LE VERBEXIT CEEERRIP.

Application domain (AP)

- **Verbx dfhpd660 'ap=3'**
  - FIND on keyword 'violation' to see any storage violations currently detected by CICS for all task storage.

```
USER31.19984 1E317490 USER storage above 16MB

0000  40404040 40404040 40404040 40404040  40404040 0000000C 0000000C 0000000C *                       ............*
0020  00000000 0C000000 0C000000 0C000000  0C000000 0C00000C 0000000C 00000C00 *................................*
0040  00000000 0C00000C 00000C00 000C4040  40404040 40404040 40404040 40404040 *..............                  *
0060  40404040 40404000 00000C00 00000C00  00000C00 00000C00 000C0000 0C00000C *       ........................*
0080  00000C00 00000C40 40404040 40404040  40404040 40404040 40404040 40404040 *.......                         *
00A0  40404040 40000000 0C000000 0C000000  0C000000 000C0000 000C0000 000C0000 *     ..........................*
00C0  000C0000 000C0000 0C000000 0C00000C  00000000 000C0000 0C00000C 00000C40 *............................... *
00E0  40404040 40404040 40404040 40404040  40404040 40404040 0000000C 0000000C *                        ........*
0100  0000000C 0000000C 00000C00 000C0000  0C00000C 0000000C 40404040 40404040 *........................        *
0120  40404040 40404040 40404040 40404040  40404040 40400000 000C0000 000C0000 *                          ..........*
0140  000C0000 00000C00 00000C00 00000C00  00000C00 00000C00 000C0000 000C0000 *................................*
0160  0C000000 00000C00 000C0000 0C00000C  40404040 40404040 40404040 40404040 *................                *
...
4440 -    48DF LINES SAME AS ABOVE
48E0  40404040 40404040 E4F0F0F1 F9F9F8F4                                     *        U0019984                *

** DFHPD0124  Storage violation detected at 1E317490. Leading SAA is invalid.
```

  - The beginning of this storage should have a check zone, U0019984, matching the trailer which is intact.

Find the most recent EXEC CICS request (AP)

- **Verbx dfhpd660 'aps=<taskid=xxxxx>**
- Page forward to find the System EIB, EIUS, User EIB.

```
SYSEIB.44444 0005A494 System EXEC Interface Block

-0008                                                   5CE2E8E2 C5C9C240 *                    *SYSEIB *
0000  0141919C 0110130F C2C9C7C4 0044444C  C1D6D9F1 000002D5 00807C06 02000000 *..j....BIGD....AOR1...N..@....*
0020  00000000 00000000 00000000 00000000  00000040 40404040 40404000 00000000 *.................         .....*
0040  00000000 00000000 00000000 00000000  00000000 00                        *.....................           *

EIUS.44444 00102008 EXEC Interface User Structure

0000  00B46EC4 C6C8C5C9 E4E24040 40404040  17E00008 00000000 17E03950 00000000 *..>DFHEIUS      .\.....\.&....*
0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000 *................................*
0040  00000000 00000000 001020D0 17E03878  00102418 00000000 00000000 00000000 *.........}.\...................*
...

EIB.44444 001020D0 EXEC Interface Block

-0010                                         00656EC4 C6C8C1D7 6DC4C6C8 C5C9C25C *              ..>DFHAP_DFHEIB**
0000  0141919C 0110130F C2C9C7C4 0044444C  C1D6D9F1 000002D5 00807C06 04000000 *..j....BIGD....AOR1...N..@....*
0020  00000000 00000000 00000000 00000000  00000040 40404040 40404000 00000000 *.................         .....*
0040  00000000 00000000 00000000 00000000  00000000 00                        *.....................           *
```

- The System Exec Interface Block (EIB) +x'1B' (2 bytes) contains the function code for the most recent EXEC CICS request. e.g. **0602**

The most recent EXEC CICS request (AP)

- The description of the EIB function codes (field EIBFN) can be found searching in the CICS InfoCenter (or CICS Application Programming Reference manual).
  - Search on keyword EIBFN and the 2-byte code:

Code	Command
0602	READ
0604	WRITE

- If the EIB function code in the (User) EIB is different than the function code in the System EIB, then typically the current EXEC CICS request has not yet completed.
  - The function code in the System EIB is the most current request.
  - The current EXEC CICS request in the example is an EXEC CICS READ FILE. The previous request was a WRITE FILE.

Who issued the last EXEC CICS request ?

- **Verbx dfhpd660 'aps=<taskid=xxxxx>'**
- FIND the EIUS. **EIUS +x'50'** contains the address of the Register Save Area (RSA) for the program issuing the EXEC CICS request.

```
EIUS.44444 00102008 EXEC Interface User Structure

0000  00B46EC4 C6C8C5C9 E4E24040 40404040  17E00008 00000000 17E03950 00000000 *..>DFHEIUS      .\.......\.&....*
0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000 *................................*
0040  00000000 00000000 001020D0 17E03878  00102418 00000000 00000000 00000000 *...........}.\................*
...
```

- The registers are saved in the RSA at offset x'C' in the order Reg14 – Reg12. These will be the application's registers at time of the EXEC CICS request.
  - Reg14 will point into the requesting program, immediately after the EXEC CICS request.

```
RSA:
00102418                          00000000   17E0B828   |         .....\.. |
00102420   00000000   15F1DAC0    00000000   17E1068C   | .....1.{........ |
00102430   18A46060   97E0C0D8    18910DA0   17E10618   | .u--p\{Q.j...... |
00102440   189109A8   189109E8    17936440   17F01680   | .j.y.j.Y.l. .0.. |
00102450   17E0BA28   001020D0    17F0EA38   15F1D928   | .\.....}.0...1R. |
```

- Identify the requesting module using the Loader domain, the Program domain, or backing up in storage looking for a module ID.

Trace (TR)

- **Verbx dfhpd660 'tr=1'**
  - Formats abbreviated CICS trace entries.

Trace sequence number

```
51392 QR  AP 00E1 EIP   ENTRY READ                                        0004,1EA3C158 .tA.,09000602 .... =000050=
51392 QR  AP 04E0 FCFR  ENTRY READ_SET           FILE1  ,00000000,9EA41D63,NO,EQUAL,FCT_VALUE,KEY,NO,NO       =000051=
51392 QR  DD 0301 DDLO  ENTRY LOCATE             1E95D820,1E5E0E91,CALLER,FCT,FILE1                          =000052=
51392 QR  DD 0302 DDLO  EXIT  LOCATE/OK          00000000 , 1EC715B8                                         =000053=
51392 QR  AP 04E1 FCFR  EXIT  READ_SET/EXCEPTION FILE_DISABLED,0,00000000,0,00000000,,,NO                   =000054=
51392 QR  AP 00E1 EIP   EXIT  READ               DISABLED                  00F4,00000032 ....,00540602 .... =000055=
```

- **Verbx dfhpd660 'tr=2'**
  - Formats full CICS trace entries. Time is Local time.

```
AP 00E1 EIP ENTRY READ                              REQ(0004) FIELD-A(1EA3C158 .tA.) FIELD-B(09000602 ....)

      TASK-51392 KE_NUM-0080 TCB-QR   /009BD690 RET-A09322AC TIME-01:37:38.4093793127 INTERVAL-00.0000032500      =000050=

AP 04E0 FCFR ENTRY - FCN(READ_SET) FILE_NAME(FILE1) ENVIRONMENT_IDENTIFIER(00000000) RECORD_ID_ADDRESS(9EA41D63) GENERIC(NO)
             KEY_COMPARISON(EQUAL) READ_INTEGRITY(FCT_VALUE) RECORD_ID_TYPE(KEY) CONDITIONAL(NO) BYPASS_SECURITY_CHECK(NO)

     TASK-51392 KE_NUM-0080 TCB-QR   /009BD690 RET-9D9FE692 TIME-01:37:38.4093797189 INTERVAL-00.0000004062      =000051=
        1-0000  00880000 00000038 00000000 00000000  B40BC12C 7C800000 02000100 00000000  *.h.................A.@.........
          0020  00000000 C6D0D3C5 F1404040 00000000  00000000 00000000 00000000 00000000  *....FILE1   ....................
          0040  00000000 00000000 00000000 00000000  00000000 00000000 9EA41D63 00000000  *............................u.....
          0060  00000000 00000000 00000000 00000000  00000000 00000202 01000002 01020002  *................................
          0080  00000200 00000000                                                        *........
```

- **Verbx dfhpd660 'tr=3'**
  - Formats abbreviated trace entries followed by full trace entries.

Short trace entries & Exception entries (TR)

- **Verbx dfhpd660 'trs=<short>'**
- Short trace entries contain the same information as the abbreviated trace entries plus the following information from the full trace entries:
  - Interpreted parameter list, showing keyword and value.
  - Return address
  - Time stamp / Interval

```
41394 QR AP 00E1 EIP  ENTRY  READ                          REQ(0004) FIELD-A(1EA3C158 .tA.) FIELD-B(09000602 ....)
                                                                    RET-A09322AC 01:37:38.4093793127 00.0000032500  =000050=
41394 QR AP 04E0 FCFR FCN(READ_SET) FILE_NAME(FILE1) ENVIRONMENT_IDENTIFIER(00000000) RECORD_ID_ADDRESS(9EA41D63) GENERIC(NO)
               KEY_COMPARISON(EQUAL) READ_INTEGRITY(FCT_VALUE) RECORD_ID_TYPE(KEY) CONDITIONAL(NO) BYPASS_SECURITY_CHECK(NO)
                                                                    RET-A09322AC 01:37:38.4093793127 00.0000032500  =000051=
```

- CICS writes Exception trace entries for most errors and abnormal conditions. These exception entries cannot be suppressed, and are written even if CICS tracing is stopped.
  - FIND on ***EXC** to see them:

```
53069 QR  AP 1942 APLI  *EXC* Program-Check  START_PROGRAM,PROGRAM1,CEDF,FULLAPI,EXEC,NO,00000000 =000066=
```

Formatting selected tasks and entries

- Verbx dfhpd660 'trs=<taskid=xxxxx,abbrev|full|short>'
  - Formats trace entries for the selected task only.
    - e.g. **verbx dfhpd660 'trs=<taskid=12345,full>'**
  - Can select multiple tasks:
    - e.g. **verbx dfhpd660 'trs=<taskid=(xxxxx,yyyyy),abbrev>'**

- Verbx dfhpd660 'trs=<entry_num=(nnnnnn-nnnnnn)>'
  - Formats trace entries selected by trace sequence numbers.
  - If you encounter message DFHPD0123 for an error/program check during trace formatting, trace formatting stops. You can format the remaining trace entries if you bypass the failing one:
  - e.g. **verbx dfhpd650 'trs=<entry_num=xxxxxx-999999,full>'**
    - where xxxxxx is the last 'good' trace entry +2 to get around the entry in error.

Additional trace options

- ‘trs=<ke_num=(nnnn,nnnn)>’
  - List entries for a specific Kernel number(s).
- ‘trs=<termid=tttt>’
  - List entries pertaining to a specific Terminal ID.
- ‘trs=<timerg=(hhmmss-hhmmss)>’
  - List entries from a specific time range.
- ‘trs=<tranid=tran>’
  - List all trace records related to a transaction ID.

- Note: The full list of additional trace options can be found in the CICS InfoCenter – see Trace Selection Parameters