CICS TS for z/VSE Design and Best Practice

Mike Poil
michaelalanpoil@gmail.com

Revision date: 7th April 2021

Important Disclaimer

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

Introduction

This is relevant for all CICS TS releases on z/VSE and is referred to as "CICS". If "z/OS" is mentioned, it normally relates to CICS TS for z/OS. When CICS was released in 1999, OS/390 was the name of what is now z/OS. EXEC DFHSIP, . . , OS390" allows CICS to use the emulated SVC/PC services in z/VSE.

The presentation looks at some of the concepts and design of CICS and provides some suggestions on how to obtain a more robust and efficient environment, however, *all* descriptions of the CICS design have been simplified.

In the March 2021 version, I added much more about the CICS Dispatcher.

Please bear in mind that even with an understanding of the CICS design, being able to visualize exactly what is happening in the extremely dynamic CICS workload environment will often be very difficult.

Tools like BOOSTMRO and enhanced versions of old presentations are here:

http://www.vmworkshop.org/mikepoil/index.shtml

Agenda

- The CICS TS for z/VSE subtasks
- CICS Operation Overview
- Code execution on QR
- The CICS Dispatcher
- The CICS Storage Manager
- CICS Web Support
- Inter-System Communication (ISC)
- Multi-Region Operation (MRO)
- Bonus Material

The CICS TS for z/VSE Subtasks

- CICS comprises a set of z/VSE subtasks, where a subtask is a separately-dispatchable unit of work to the z/VSE Dispatcher, and dispatched according to the z/VSE design and prioritisation rules plus the user-defined PRTY.

- The Quasi-Reentrant (QR) subtask runs almost all the CICS and application program code - it is vital that it runs efficiently, and the CICS design requires it *not* to be too busy relative to cpu capacity for best performance.

- To avoid QR being blocked by a synchronous z/VSE wait, requests like OPEN and CLOSE take place on a different subtask.

- The SL (Socket Listener), SO (Socket I/O) and S8 (SSL) subtasks handle TCPIPSERVICE I/O activity, much of which is handled asynchronously, however, GIVESOCKET and CLOSE are *synchronous* and can cause (serious) problems if they either do not complete in a timely fashion or hang.

The CICS TS for z/VSE Subtasks

```
status g1
AR 0015 S0069-G1 EVA10MST 82 WAITING FOR I/O, OR ECB POSTING          Console interface manager
AR 0015    TCB=002CE61C TIB=002CE5A0 SAV=005052A0
AR 0015 S006B-G1 DFHEVID2 82 WAITING FOR I/O, OR ECB POSTING          Auxiliary trace open/close
AR 0015    TCB=002CEBBC TIB=002CEB40 SAV=002D5100
AR 0015 S006C-G1 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING          RO (program load etc.)
AR 0015    TCB=002DB07C TIB=002DB000 SAV=002D5180
AR 0015 S006A-G1 DFHEVID1 82 WAITING FOR TIMER INTERRUPT              QR is always the 4th subtask
AR 0015    TCB=002CE8EC TIB=002CE870 SAV=002D5080
AR 0015 S006D-G1 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING          SL (TCPIP=YES)
AR 0015    TCB=002DB34C TIB=002DB2D0 SAV=002D5200
AR 0015 S0077-G1 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING          SO (TCPIP=YES)
AR 0015    TCB=002FC34C TIB=002FC2D0 SAV=002D5700
AR 0015 S0074-G1 DFHEVID0 82 WAITING FOR I/O, OR ECB POSTING          Journal open/close
AR 0015    TCB=002E78EC TIB=002E7870 SAV=002D5580
AR 0015 S0070-G1 DFHSKTSK 82 WAITING FOR I/O, OR ECB POSTING          File open/close etc.
AR 0015    TCB=002DBBBC TIB=002DBB40 SAV=002D5380
AR 0015 S006F-G1 DFHIRPST 82 WAITING FOR I/O, OR ECB POSTING          DFHIRP (ISC=YES)
AR 0015    TCB=002DB8EC TIB=002DB870 SAV=00504F80
AR 0015 S006E-G1 DFHEVID1 82 WAITING FOR TIMER INTERRUPT              SZ (FEPI=YES)
AR 0015    TCB=002DB61C TIB=002DB5A0 SAV=002D5280
AR 0015 S0072-G1 DFHEVID1 82 WAITING FOR I/O, OR ECB POSTING          S8 (SSL TCPIP=YES + KEYFILE=value)
AR 0015    TCB=002E734C TIB=002E72D0 SAV=002D5480
AR 0015 M002E G1 COMSZCCA 82 WAITING FOR I/O, OR ECB POSTING          Main task (mostly idle)
AR 0015    TCB=00259310 TIB=00259290 SAV=00500000
AR 0015    SCB=00259000 PCB=00259090 COM=00259518
AR 0015 1I40I  READY
```

- The CICS subtask order is consistent, but not all subtasks may be present; Vendor and z/VSE subtasks can also appear and typically have a higher priority than most of the CICS subtasks.

The CICS TS for z/VSE Subtasks

- z/VSE task dispatch priority high-to-low is normally as shown below, excluding Vendor subtasks, but EVA10MST's priority can vary.

- If you run BOOSTMRO, DFHIRPST has a higher priority than QR.

  - DFHEVID1 S8 SSL Socket I/O
  - DFHSKTSK
  - DFHEVID0
  - DFHEVID1 SO normal Socket I/O
  - DFHEVID1 SL Socket Listener (one task for all TCPIPSERVICEs)
  - DFHEVID1 RO
  - DFHEVID2
  - DFHEVID1 SZ
  - DFHEVID1 QR
  - DFHIRPST
  - EVA10MST
  - Main task

The CICS TS for z/VSE Subtasks

- The CICS Dispatcher only manages the "DFHEVID1" subtasks plus the z/VSE Main Task, which will be in an z/OS SVC 1 wait when they are idle - the Dispatcher does **not** use z/VSE WAITM SVC 29 nor z/VSE (I)WAIT SVC 7 as most of the CICS code uses z/OS emulation services.

- The SVC 1 wait on QR is called the "Partition Exit" wait and QR will be idle for a maximum of ICV milliseconds (the true minimum being 250), although completion of I/O etc. typically causes QR to be dispatched by z/VSE before ICV expires.

- Each of the Dispatcher subtasks is often referred to as a "TCB", which is the name used in the z/OS world for a "task" and is a z/OS control block (not to be confused with the z/VSE TCB task control block!).

- Other subtasks are managed differently, with either z/OS SVC 1 idle waits or z/VSE SVC idle waits.

CICS Operation Overview

- CICS runs work as a result of receiving external requests via VTAM or TCP/IP, or internal requests from EXEC CICS START etc.
- The request normally causes a new USER TASK to be attached with a TRANSACTION ID, and is always queued for dispatch on QR.
- When the number of active and suspended user tasks hits MXT, a new task is attached, but it is SUSPENDed in an MXT wait.
- A purge or forcepurge for a task in MXT wait is deferred until MXT is resolved and the task can be dispatched to perform the purge request.
- Tasks in MXT wait occupy some storage, and when MXT is resolved it may impact CICS due to the processing of the backlog.

CICS Operation Overview

- MXT is invisible unless it is identified by Vendor software, although MXT wait can be seen in CEMT INQUIRE TASK if you write your own monitor.
- Before applying the MXT check, the task may be subject to a Transaction Class (TCLASS) MAXACTIVE limit related to the Transaction ID.
- Purging a task in a TCLASS wait will work if CICS is **not** at MXT.
- A Purge Threshold can be set to stop a long TCLASS wait queue, for example, PURGETHRESH(4) will allow 3 tasks in a TCLASS wait with additional task attach requests failing with an AKCC abend.
- SOS will also stop a new user task being attached.

CICS Operation Overview

- CICS also attaches and runs a number of SYSTEM tasks that are not subject to a TCLASS and MXT check.

- *Best Practice: Configure CICS to avoid **unnecessary** MXT and TCLASS waits, both of which can be seen to occur in CICS Statistics data. I have sometimes seen TCLASS settings that were relevant a long time ago, but are not now. Perhaps put (TPCIPSERVICE) CWXN in a TCLASS with a Purge Threshold.*

- *A Very Important Best Practice: Always keep a CEMT task active to help you manage CICS when a new CEMT task cannot be started due to MXT or SOS.*

Code execution on QR

```
PROGA1    CSECT
          EXEC CICS STARTBR FILE('FILEA') RIDFLD(RID1) KEYLENGTH(0)     *
                GTEQ GENERIC RESP(RESP) RESP2(RESP2)
* EXEC CICS results in a "CALL" to DFHEIP.
* If no abend, DFHEIP typically returns control with response values, but
* if a wait is required, the CICS Dispatcher is invoked to SUSPEND the task. .
* This allows a different task to be dispatched, and DFHEIP will pass control
* back when the wait has completed and the task is RESUMEd and re-dispatched.
LOOP      DS   0H
          EXEC CICS READNEXT FILE('FILEA') INTO(REC) RIDFLD(RID2)       *
                RESP(RESP) RESP2(RESP2)
          CLC  RESP,DFHRESP(ENDFILE)
          JE   ENDBR
          CLC  RESP,DFHRESP(NORMAL)
          JE   LOOP
          EXEC CICS ABEND ABCODE('MIKE')
ENDBR     DS   0H
          EXEC CICS ENDBR FILE('FILEA') RESP(RESP) RESP2(RESP2)
          EXEC CICS RETURN
```

The CICS Dispatcher

- The Dispatcher's main functions are as described below, but a review of formatted trace data with AP=1,EI=1 and DS=1 or DS=1-2 trace options can help to get a better picture of how CICS works.

1. To provide a service to allow tasks to be attached.
2. To provide services to allow tasks to suspend processing (wait) until an event completes.
3. To decide which CICS task to run next under a given TCB based on priority, but this is biased to give recently dispatched tasks the highest priority.
4. To issue an z/OS WAIT SVC 1 on a TCB when no CICS tasks are ready to run under that TCB.
5. To provide a service to allow tasks to move from one TCB to another.

The CICS Dispatcher

- A task is represented by a DTA control block in the Dispatcher Domain.
- For each TCB, there is one list of all DTAs and a second that *only* contains DTAs for tasks that are dispatchable (i.e. *waiting* for dispatch status "D" in CEMT I TA) - more than a small number indicates cpu contention.
- The Dispatchable list is kept sorted in descending Dispatch Priority order and a task's Dispatch Priority is **not** a constant value during its lifetime - Dispatch Priority become more important as the size of the Dispatchable List grows.
- The active task on a TCB has the status "R" and is not on the Dispatchable list - *only one DTA can be in in an "R" state per TCB, but obviously there can be more than 1 "R" state task at any one time as there are multiple TCBs*.
- CICS performs the ATTACH, and the task is placed on the *QR* Dispatchable list according to its just-calculated Dispatch Priority relative to others there.

The CICS Dispatcher

- EXEC CICS processing in CICS code may result in one or more DFHDSSRM SUSPEND/RESUME pairs to swap between the DTA lists.
- SUSPEND flags the DTA as Suspended and the Dispatcher will either dispatch the highest priority DTA on the Dispatch list or cause the TCB to enter an z/OS SVC 1 wait on a list of ECBs - for QR this is the "Partition Exit" wait and lasts until QR is dispatched by z/VSE because the ICV timer has expired or because another ECB was posted by z/VSE.
- RESUME flags the DTA as Dispatchable.
- Many DTAs can be added to the QR Dispatchable List at the same time, e.g. FCXCWAIT, which can result in many SUSPEND/RESUME/Dispatch cycles for many of these tasks because the highest priority task can acquire the resource, SUSPEND, and allow the next one to run only to hit the same contention problem and suspend again (repeat . . .).

The CICS Dispatcher

- There are different types of SUSPEND that can occur.
- SUSPEND: there will be an explicit RESUME and no ECB is involved.
- WAIT_EXTERNAL: CICS expects a z/VSE POST and has a special "POST Exit" invoked by the z/VSE Turbo Dispatcher when the event is complete to quickly add the DTA to the Dispatchable list.
- WAIT_OLDW and WAIT_OLDC: *The predominant type of QR SUSPEND*. A z/VSE POST for an ECB may or may not occur and all of these ECBs are classified as "Hand Postable"; the Dispatcher must check them for completion either just before the QR Partition Exit wait, or at periodic intervals in the Dispatcher scan.
- The Dispatcher maintains a third list of DTAs that are SUSPENDed for a Hand Postable ECB.

The CICS Dispatcher

- See the CICS TS for VSE/ESA Problem Determination Guide page 69 onwards and the CICS TS for z/VSE 2.n Enhancements Guide for task SUSPEND resource type and name pairs etc.

- CICS is at a disadvantage compared to z/OS, which uses a variation of WAIT_EXTERNAL that is *very* responsive to event completion and is used for a high percentage of the total DFHDSSRM waits - a Local Fix at the end of the presentation *might* help to mitigate against this disadvantage.

- *Best Practice: ICV=250 can help wait completion to be handled faster when CICS is idle in the Partition Exit wait.*

The CICS Dispatcher

- A task's dispatch priority is **not** a constant, instead it has one of two values, partly depending on whether Priority Aging (SIT PRTYAGE=n) is active or not.
- Irrespective of PRTYAGE, the aim is to give every dispatched task a high Dispatch Priority unrelated to the defined priority, which it can retain for a time while suspended.
- With PRTYAGE=0 and when no longer at the high priority, the Dispatch Priority is the same as the defined priority.
- With PRTYAGE>0 and when not at the high priority, the Dispatch Priority **decreases** over time (this is **not** as described by CICS documentation!).
- The dynamic nature of the workload makes this design very difficult to visualise outside of a dump of a busy system formatted with AP=1 and DS=3.
- There is a ZAP in the Bonus Material that changes some of this design.

The CICS Dispatcher

- *Please remember that the Dispatcher is not normally invoked until the active task is suspended, hence a low priority transaction can (significantly) impact higher priority transactions if it does not suspend regularly!*

- EXEC CICS SUSPEND and EXEC CICS CHANGE TASK PRIORITY(n) cause the Dispatcher to be invoked, which results in the task's Dispatch Priority being re-assessed relative to the others on the QR Dispatchable list, *potentially* allowing a higher priority task to be dispatched - a SUSPEND "wait" does not occur!

- 100 consecutive EXEC CICS VSAM file requests without another task performing any will result in an implicit EXEC CICS SUSPEND.

- *Best Practice: Define Transaction Priority according to performance objectives, but do not use a high priority that conflicts with CICS system transactions. It is very difficult to determine if PRTYAGE is of any real value never mind what is the "best" value to use!*

The CICS Dispatcher

- To avoid an instruction loop blocking QR, CICS will attempt to cause an AICA abend after ICVR+500 milliseconds of *cpu* time without a task wait, but it does not always work.
- *Best practice: Use SIT ICVR=500, with higher values for RUNAWAY(n) in specific CSD Transaction definitions as required.*
- If QR requires a different Dispatcher TCB to do something, it issues a CHANGE_MODE request, and the task is added to the Dispatchable list of the other TCB.
- A CHANGE_MODE in this subtask will move the task back to the QR TCB Dispatchable list, where it may need to wait to be dispatched again.
- However, the second subtask could perform another CHANGE_MODE etc.

The CICS Storage Manager

- There are z/VSE LVCs that have much more detail on this subject.
- SIT DSALIM and EDSALIM define the maximum amounts of storage for CICS Storage Management (SM) to use.
- Partition Getvis for the DSALIM and EDSALIM values is allocated during CICS initialisation, and it remains allocated even if not all used by SM.
- If the whole of DSALIM is used, SOS Below becomes a possibility, similarly for EDSALIM and SOS Above.
- *Best Practice: Use CICS Statistics to check (E)DSALIM usage and increase the allocations as required but be careful because over-allocating (E)DSALIM could cause Partition Getvis issues! Either can cause a CICS outage. Avoid 24-bit DSALIM use wherever possible.*

The CICS Storage Manager

- (E)DSALIM storage is dynamically suballocated to the DSAs (as described on the next slide) in EXTENTs of 256K below 16MB and 1MB above 16MB; once allocated to a DSA an extent normally remains allocated even if all of the extent storage is freed.

- As CICS approaches SOS, it tries to avoid it by iteratively freeing storage belonging to unused programs ("Program Compression"), reducing new task priority to reduce concurrent usage, and completely free extents are passed to the DSA that needs them; you can see Program Compression activity in CICS Statistics output from DFH0STAT and DFHSTUP.

- *(E)DSALIM can be increased by CEMT in units of 256K/1MB while CICS is active if free contiguous Partition Getvis is available .*

- *Best practice: Use CICS Program and Storage Statistics to warn you about the possibility of an SOS so that you can take action **before** it occurs.*

The CICS Storage Manager

- The types of DSA are:
  - UDSA: 24-bit User-key task (lifetime) storage.
  - CDSA: 24-bit CICS-key task storage, CICS control blocks and CICS-key phases.
  - SDSA: 24-bit GETMAIN SHARED and User-key phases.
  - RDSA: 24-bit SVA-eligible phases that are loaded into the Partition.
  - ExDSA versions are mapped into 31-bit EDSALIM storage.
- *Best Practice: Use SIT STGPROT=YES, RENTPGM=PROTECT and CMDPROT=YES. LNKEDT re-entrant application programs with "PHASE name,*,SVA" to use the (E)RDSA and protect the code from overlays that cause obscure abends, using SYSIPT PRVMOD overrides to avoid messages about the SVA copy of the program not being available. Load DFHSIP31 into the SVA to both protect it and get better cache performance from using a single copy of the large phase - please note that SIT SVA=YES is **not** required to execute DFHSIP31 in the SVA.*

The CICS Storage Manager

- CICS uses additional Partition Getvis for:
  - The Internal Trace Table (SIT TRTABSZ).
  - Reentrant CICS code Kernel Stack (variable).
  - Storage Manager (SM) control blocks (variable).
  - The transaction trace table while the transaction dump is produced (SIT TRTRANSZ).
  - Other things, but the usage is normally low.
  - Most of this is 31-bit storage.
- *Best Practice: Remember to use GETVIS xx,RESET after CICS initialises. Keep maybe 256K to 512K of free 24-bit, and at least 10MB of free 31-bit (but more if possible), increasing the Partition allocation as required. If all the 31-bit storage is used, z/VSE will try to allocate a 31-bit request in 24-bit storage!*

CICS Web Support (CWS)

- A CICS-managed Socket interface to CICS applications, not to be confused with EZASOCKT, and is the base for the SOAP and RESTful interfaces from z/VSE, and for the CICS Transaction Gateway etc.
- CICS provides **one** Listener via transaction CSOL, which "Listens" on **all** of the open TCPIPSERVICE definitions.
- CICS uses z/VSE-provided interfaces for Socket and SSL requests, and CICS TS for z/VSE 2.2 can now use Open SSL.
- CSOL executes Socket I/O on the SL subtask.
- CICS TS for z/VSE 2.2 supports HTTP 1.1.

CICS Web Support (CWS)

- Synchronous and Asynchronous Socket I/O is handled either by the SO subtask, or by the S8 subtask for CICS-managed SSL before CICS TS for z/VSE 2.2.

- Asynchronous socket I/O allows the use of DFHDSSRM task waits.

- A long-running synchronous socket request like CLOSE and GIVESOCKET can temporarily impact performance, and in extreme cases, can cause a CICS (partial) hang because the SO subtask becomes blocked - this is never a CICS problem - talk to your TCP/IP provider.

Inter-System Communication (ISC)

- CICS uses VTAM to transmit data between 2 CICS systems in the same or different mainframes for remote DPL, START, Function Shipping or APPC-based applications, using CSD CONNECTION and SESSION definitions.

- *Best Practice: ISC may provide acceptable performance if the level of activity is **not** high.*

Multi-Region Operation (MRO)

- MRO (DFHIRP) transmits data between two CICS systems in the same z/VSE system using CONNECTION and SESSION definitions.

- A very simplified data flow where CICS A is the client:
  - CICS A: Terminal Control Program (TCP) request to interact with CICS B. Allocate SVA buffers and copy the data, this side's task waits. Issue the "Switch" (send) to wake up the CICS B DFHIRPST subtask.
  - CICS B: DFHIRPST "Pull" (receive). Copy the data, attach/post this side's task and free the SVA buffers.
  - CICS B: A later TCP request to send to CICS A. Allocate SVA buffers and copy the data, this side's task waits. "Switch" to wake up CICS A DFHIRPST subtask.
  - CICS A: DFHIRPST "Pull". Copy the data, post the waiting task and free SVA buffers.

- *A maximum of 256K of 24-bit System Getvis can be used for data transfer, which is the total for **all** CICS jobs in the z/VSE system - AZI2 abends occur if the 256K limit is exceeded and is a performance issue.*

Multi-Region Operation (MRO)

- *Best Practice: High volume MRO activity has a **very significant** cpu overhead, a lot of which is due to overhead in z/VSE SVCs. Use MROLRM=YES in the Server CICS (e.g. the FOR) to reduce the cpu overhead by up to 30%. The use of MROFSE=YES is not recommended. Ensure that QR in the Client and Server CICS systems is not too busy or MRO I/O wait times might be impacted - however, using my BOOSTMRO tool to counteract this issue has had good feedback from one customer that uses a huge amount of Function Shipping. Use CICS Statistics to look for (MRO and ISC) Session allocation issues that indicate too few Sessions are available. Consider activating a Connection QUEUELIMIT and, optionally, MAXQTIME to avoid long queues forming in another CICS system when there is a problem in this CICS system ("Sympathy Sickness").*

Bonus Material

The first ZAP is for CICS TS for z/VSE 2.2 customers who would like the CICS task Dispatch Priority to be consistently set to the Defined Priority (0 to 255).

For it to work as expected, SIT PRTYAGE=0 **MUST** also be set, which may require a COLD start. To have excluded all the PRTYAGE effect would have required a much more complex ZAP. If you use PRTYAGE>0 with this ZAP, long running and important high priority tasks will see a gradual decrease in their Dispatch Priority over their lifetimes.

Without this ZAP, the Dispatcher gives recently dispatched tasks the highest Dispatch Priority no matter what their defined priorities are. Now, even a difference of 1 *might* have an effect. If it doesn't work for you, UNDO the APAR fix and re-link DFHSIP31. It is not an official "fix" and you use it at your own risk.

The second ZAP might help to resume tasks faster and hence reduce response time when they are waiting on Hand Postable ECBs (WAIT_OLDW and WAIT_OLDC). The same conditions apply as for the first ZAP.

Bonus Material

```
// JOB MSHP
/*
/* Eliminate the bias towards giving recently
/* dispatched tasks a very high priority.
/*
/* Use the same offset for CICS 2.1 5655-VSE-00-B2P
/* Use offset 0B84 for CICS 1.1 5648-054-00-B0P
/*
// EXEC MSHP
CORRECT 5655-VSE-00-B3P:DS00001 REV
AFF MOD=DFHDSTCB
ALTER 000B5C D20770482170:070007000700
RESOLVES 'ALWAYS USE THE DEFINED PRIORITY'
/*
// OPTION CATAL
// LIBDEF PHASE,CATALOG=PRD2.CONFIG
INCLUDE DFHSIP31
// EXEC LNKEDT
/&
```

Bonus Material

```
/* The SCAN_HAND_POSTABLE periodic check in the Dispatcher scan is
/* changed from 100 to 25 milliseconds. This might cause CICS to
/* RESUME tasks faster and reduce overall response time.
/* Feel free to experiment with other values.
/*
/* Use the same offset for 5655-VSE-00-B2P, but use offset 002038
/* for 5648-054-00-B0P.
/*
// EXEC MSHP
 CORRECT 5655-VSE-00-B3P:DS00002 REV
 AFF MOD=DFHDSDM
 ALTER 002004 00000064:00000019
 RESOLVES 'SCAN HAND-POSTED ECBS MORE OFTEN'
/*
// OPTION CATAL
// LIBDEF PHASE,CATALOG=PRD2.CONFIG
  INCLUDE DFHSIP31
// EXEC LNKEDT
```

Bonus Material

```
DTA 12D73180 DISPATCHER TASK AREA
   0000  FFFFFFFF 03820001 00000000 E2E3D4C5  D4F0F240 40404040 40404040 C6C3C9D6  *.....b......STMEM02         FCIO*
                                    Current/Residual Suspend Type ---------------
   0020  E6C1C9E3 00000000 FE000000 12D73080  00000000 00000000 FFFFFFFF FFFFFFFF  *WAIT.........P..................*
         ------->
   0040  80000000 03FF0000 26AC8296 F2C2E480  D9527D6D D1F11B80 00000000 00000000  *.........bo2BU.R.'_J1..........*
                          Dispatch_priority
                 03 = Dispatchable wait
                 04 = Running
                 FC = Suspended
   0060  00E9CE00 09EE4400 01020001 00000000  050EF601 03010003 12D8F000 09EE98D4  *.Z.............6......Q0...qM*
                                         ECB/ECBLIST: low-order 1 = z/VSE ECB format and is not part of the address
                                                      01 ECB address
                                                      02 ECBLIST address
   0080  0A60C700 0005862C 00000000 00000000  FFFF0004 C4B40000 00000000 00000000  *.-G...f.............D...........*
                 Task #
   00A0  00000000 00000015 00000003 0A60C700  00000000 00000000 00000000 00000000  *.............-G.................*
   00C0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
   00E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
```

Trademarks

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

BladeCenter*
DB2*
DS6000*
DS8000*
ECKD
FICON*
GDPS*
HiperSockets
HyperSwap
IBM LinuxONE Emperor
IBM LinuxONE Rockhopper
IBM Z*
IBM z13*
IBM z14*
IBM z15*
OMEGAMON*
Performance Toolkit for VM
Power*
PowerVM
PR/SM
RACF*
Storwize*
System Storage*
System x*
System z*
System z9*
System z10*
Tivoli*
zEnterprise*
z/OS*
zSecure
z/VM*
z Systems*

* Registered trademarks of IBM Corporation

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
OpenStack is a trademark of OpenStack LLC. The OpenStack trademark policy is available on the OpenStack website.
TEALEAF is a registered trademark of Tealeaf, an IBM Company.
Windows Server and the Windows logo are trademarks of the Microsoft group of countries.
Worklight is a trademark or registered trademark of Worklight, an IBM Company.
UNIX is a registered trademark of The Open Group in the United States and other countries.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

This information provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g., zIIPs, zAAPs, and IFLs) ("SEs"). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the “Authorized Use Table for IBM Machines” provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html (“AUT”). No other workload processing is authorized for execution on an SE. IBM offers SE at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.