

Setting the Interval Control Values in CICS

CICS has three interval control values that are specified in the System Initialization Table (SIT), via overrides or through the use of CEMT command to set a new value. Setting these parameters correctly can help reduce system CPU overhead and provide a better use of resources. This is especially true in installations that have many CICS regions. This article explores these parameters and provides recommendations as to what values should be used. The three parameters and a brief description are as follows:

- Interval Control Value (ICV) – used to specify the maximum amount of time CICS is to wait when there is no work to do, that is, no ready task to dispatch within CICS. The time is given in milliseconds and the default time is 1000 ms. or one second
- Interval Control Value for Runaway Tasks (ICVR) –used to specify the maximum amount of real CPU time that is allotted to a task before the task has to return control to CICS or else the task is cancelled with a cancel code of AICA. The time is given in milliseconds and the default time is 5000 ms. or five seconds
- Interval Control Value for Terminal Scan Delay – used to specify the amount of time that is to elapse before CICS deals with some terminal I/O activity. The time is given in milliseconds and the default is 500 ms. or half a second

Let us address the ICV parameter first. When CICS has no transaction ready to dispatch or work to do, CICS enters a wait state based on an ECB list that includes the ICV timer. Any interrupt that posts an ECB completion wakes CICS even if the ICV timer ECB has not completed. Traditional recommendations specify that the ICV can be used to protect the working set of the CICS region. Periods of low terminal or transaction activity can represent a problem for CICS if the system is under pressure for real storage. Setting the ICV to a low value such as 1000 ms. (1 second) will wake up CICS more often to try and protect part of the nucleus. However, waking up CICS this often to find nothing to dispatch and to be placed into a wait state again can be a waste of CPU resources.

It is important to note that CICS is a very disperse on-line system that requires a lot of real storage to execute. During normal transaction execution many CICS management programs, control blocks and tables must be accessed in addition to the user program and storage areas. So, waking CICS often as a result of a low ICV only protects a very small portion of the total CICS code and areas. In addition, due to increased real storage availability on modern processors, the need to protect a small amount of CICS storage may not be effective trade-off against CPU cycles and at the expense of lower priority regions. This is especially true when there is a period of very low transaction activity.

ICV like other SIT parameters tend to be “inherited”, that is, they were defined many years ago and are never adjusted. The conditions under which these parameters were set many years ago have changed. Faster CPUs, improved I/O

devices and more real storage availability are some of the changes that have occurred. Yet, the original specification for these SIT parameters remains the same. In addition, the number of CICS regions has grown significantly. It is not unusual to see a test CICS, model office or quality CICS and a production system combination in many installations. In many cases, this type of CICS regions can be set up by application, proliferating the total number of CICS systems in a CPU complex. It is not unusual to see more than 100 CICS systems in a z/OS or OS/390 installation. Therefore, having all these CICS systems, especially test and model office or quality CICS systems, with a low ICV amount may be questionable.

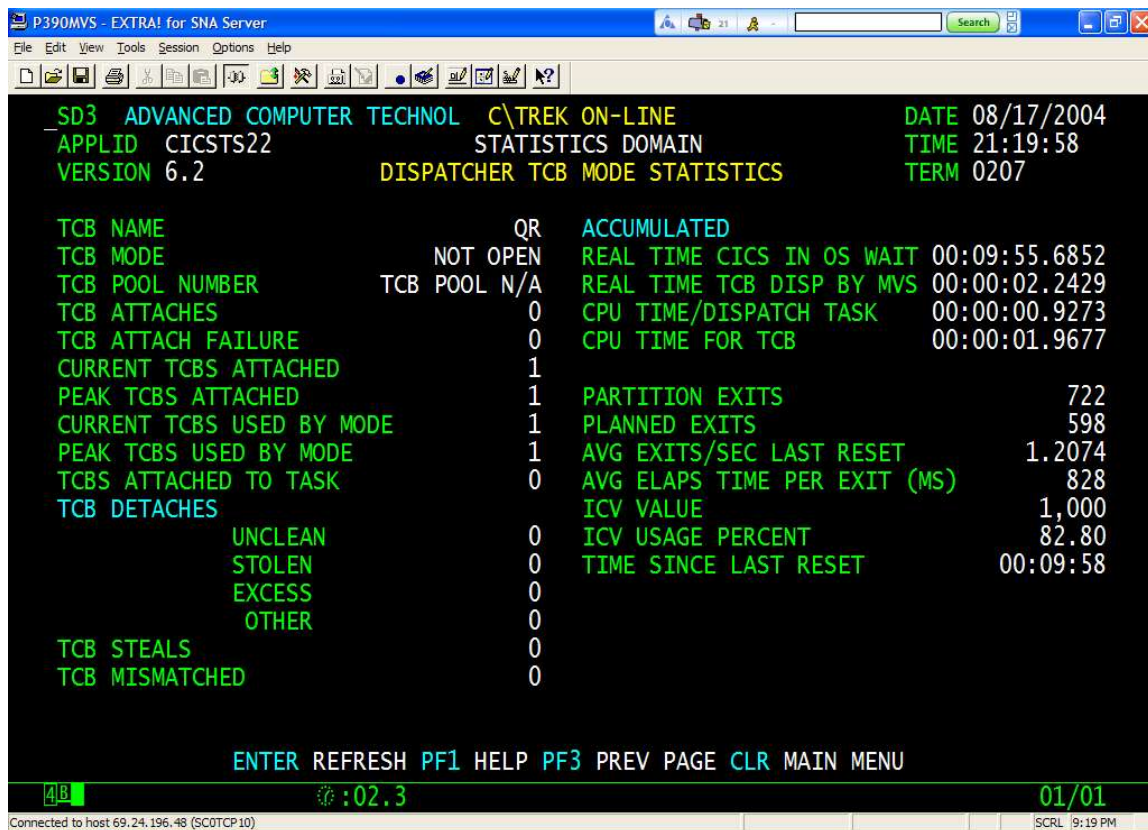


Figure 1. CPU Time for ICV=1000 (1 Second)

As an example, we changed the CICS statistics collection period from 3 hours to ten minutes so that we could get the information and reset the statistics quickly. We ran two tests, one with an ICV=1000 (1 second) and one with an ICV=10000 (10 seconds). Figure 1 shows the results of the first test with an ICV of one second. The average length of the ICV interval for this test was 828 milliseconds and the TCB CPU time was 1.9677 seconds. The figures include the transaction to capture the data in both examples.

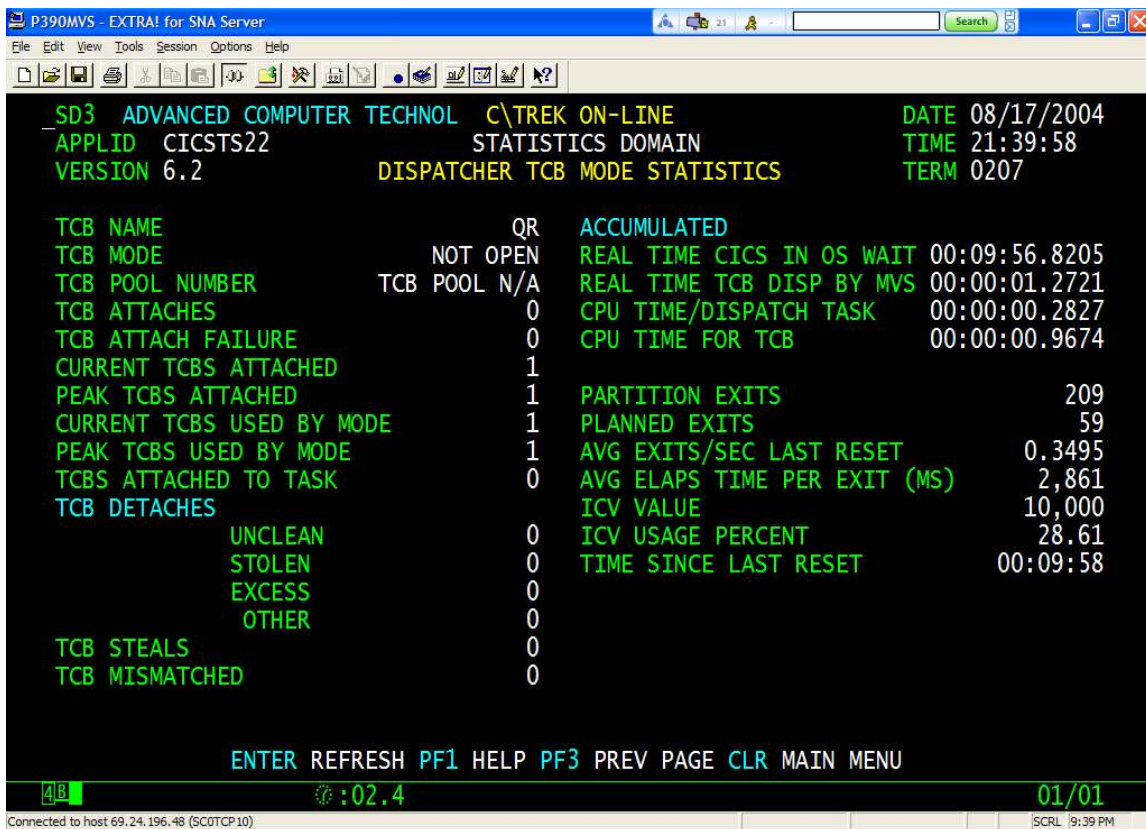


Figure 2 CPU Time for ICV=10000 (10 Seconds)

Figure 2 reflects the second test when the ICV was set to ten seconds. The results were an average ICV of 2,861 milliseconds and the TCB CPU time was .9674 seconds. Again as in the first benchmark, the figures also include the transaction overhead used to capture the information. Increasing the ICV to 10 seconds represented a savings of one CPU second or just over 50% CPU overhead. The one second difference over the measurement period of ten minutes represented around .2% overhead of the overall CICS CPU used. This savings would represent a 1% CPU savings for every five CICS systems that have a low ICV such as 1000 milliseconds. A lower ICV value requires more CPU overhead. As with all benchmarks, your mileage may vary.

The setting of the ICV to a value between five to ten seconds should reflect lower CPU overhead and would result in a positive factor for lower priority regions. It should be noted that during normal execution, the system would probably respond to other interrupts (e.g., VSAM I/O) than to the ICV timer value, especially in very active systems. There may be a case where you may want to consider a lower ICV value. The use of MROBTCH value greater than one (e.g., 2) may require a lower ICV setting because CICS will hold transactions in a wait queue until the batch is complete (two transactions if MROBTCH=2), the ICV value or three seconds expires, whichever occurs first. In this case, a lower ICV value would be justified to avoid an artificial delay in response time. One final area that may be affected by a high ICV value is the logon process to CICS that may have to wait the full ICV time before CICS can perform the auto-install process. Therefore, very high values such as ten minutes may not be good. ICV

values in the five to ten seconds range should be a good compromise for the logon process.

The second SIT value is the ICVR that is used to control the amount of CPU time allotted to a transaction during one dispatch for execution. The ICVR is another possible “inherited” value. The IBM default of five seconds is quite high. First of all, back in 1990 when processor speeds were in the 20 to 30 MIPS range, an ICVR of five seconds represented around 100 to 150 million instructions. Today’s processors are capable of executing a significant number of instructions in one second (e.g., 250 MIPS or higher). A high ICVR value such as five seconds represents over 1.2 billion instructions. This is an extremely long time to determine that a transaction is in a loop and to take action. In other words, a high ICVR can result in wasted CPU cycles when a task enters a loop.

A second problem associated with a high ICVR can occur with high transaction volume CICS regions. For example, imagine a region that has an MXT value of 100, an ICVR of five seconds and processes around 30 transactions per second. If a transaction falls into an instruction loop, CICS is not going to react and cancel the offending task for a full five seconds. However, by the time CICS gets around to canceling the looping transaction with an AICA, 150 transactions will have arrived to the system waiting for processing. Add the transactions that were in the system when the loop occurred that had not completed and you have a maximum task situation that can lead to other problems such as short on storage (SOS) or maximum transaction classes (TCLASS). So, selecting a value for the ICVR has to take into account the transaction rate per second, especially on high transaction volume systems. Therefore, the ICVR should be set to a value of less than one second (e.g., 600 milliseconds) to ensure that CICS can react to a looping task without endangering the system and not wasting resources unnecessarily. Should a transaction require more than the system wide ICVR, then specify the RUNAWAY value in the transaction definition for that particular transaction in Resource Definition Online (RDO) and not provide a “open door to waste cycles” for all transactions in the system.

The final timer control is the Interval Control Value for Terminal Scan Delay or ICVTSD. This timer was originally oriented to networks using the Basic Telecommunications Access Method (BTAM) to determine when polling was due on a communications line. The terminology used for this polling activity was a major and a minor scan. Once networks converted to the Virtual Telecommunications Access Method (VTAM), the method for determining terminal service changed (for the better) and this parameter did not apply to this type of network. Terminals that require service are placed on an activate queue that is anchored off the TCTFX control block. This queue is checked every time the dispatcher reviews tasks to dispatch because of the special relationship that exists with the TCP/ZCP for task CSTP. Unfortunately, the documentation regarding this parameter is sometimes contradictory and confusing and there is a tendency to not adjust it. Thus, in many installations, the ICVTSD joins the “inherited” parameter list.

Again we did the same benchmark to obtain the CPU time when the ICVTSD was set to a non-zero value versus using the default of 500 milliseconds. We

kept a high ICV value (ten seconds) to reduce timer interrupts. However, we used a five-minute interval instead of the ten minutes used for the previous ICV benchmark.

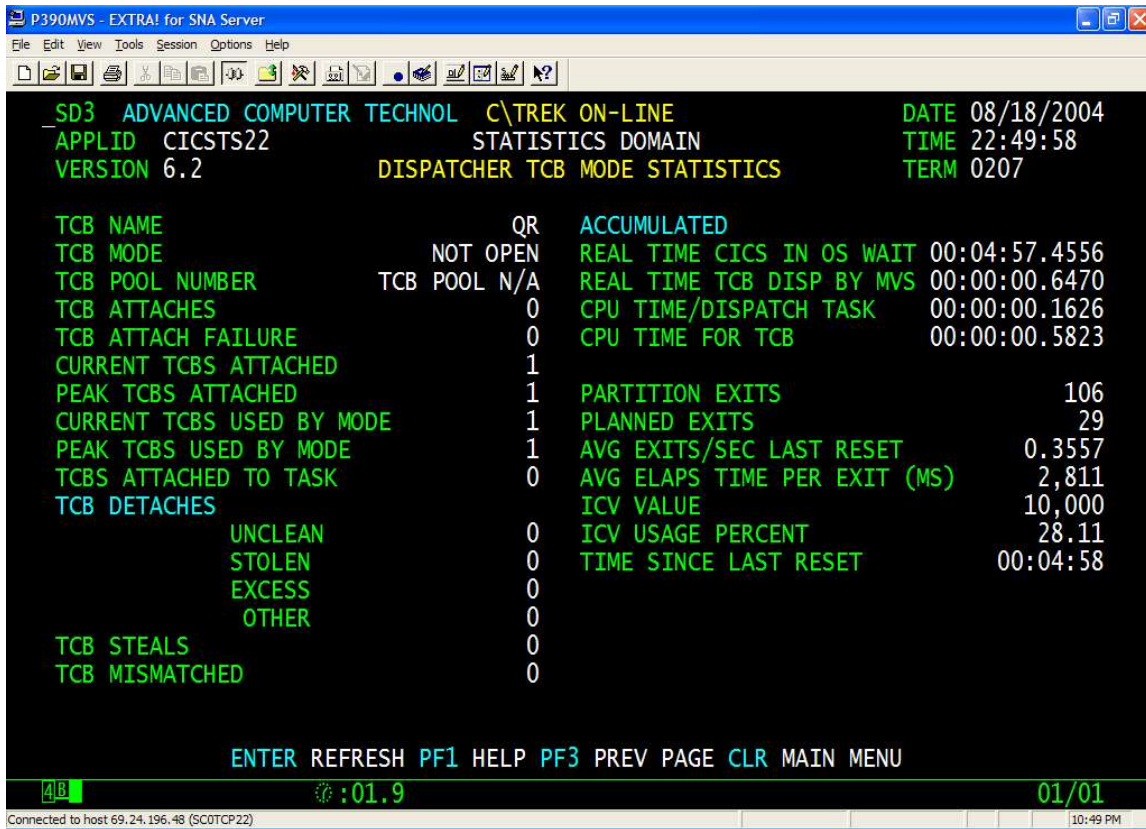


Figure 3 CPU Time Used by ICVTSD=500 and ICV=10000

Figure 3 displays the CPU when the ICVTSD is set to 500 milliseconds while Figure 4 shows the CPU when the ICVTSD is set to zero. The difference is .0649 seconds or 11% more CPU. So, if your network is VTAM oriented, then the ICVTSD should be set to zero to reduce the CPU overhead.

```

P390MVS - EXTRA! for SNA Server
File Edit View Tools Session Options Help
SD3  ADVANCED COMPUTER TECHNOL  C\TREK ON-LINE          DATE 08/18/2004
APPLID CICSTS22                STATISTICS DOMAIN      TIME 23:14:58
VERSION 6.2                    DISPATCHER TCB MODE STATISTICS  TERM 0207

TCB NAME                        QR  ACCUMULATED
TCB MODE                        NOT OPEN  REAL TIME CICS IN OS WAIT 00:04:56.9980
TCB POOL NUMBER                TCB POOL N/A  REAL TIME TCB DISP BY MVS 00:00:00.7349
TCB ATTACHES                    0  CPU TIME/DISPATCH TASK    00:00:00.1364
TCB ATTACH FAILURE              0  CPU TIME FOR TCB          00:00:00.5174
CURRENT TCBS ATTACHED          1
PEAK TCBS ATTACHED             1  PARTITION EXITS           104
CURRENT TCBS USED BY MODE      1  PLANNED EXITS             29
PEAK TCBS USED BY MODE        1  AVG EXITS/SEC LAST RESET  0.3490
TCBS ATTACHED TO TASK          0  AVG ELAPS TIME PER EXIT (MS) 2,865
TCB DETACHES                    0  ICV VALUE                 10,000
                                UNCLEAN  0  ICV USAGE PERCENT        28.65
                                STOLEN   0  TIME SINCE LAST RESET     00:04:58
                                EXCESS   0
                                OTHER    0
TCB STEALS                      0
TCB MISMATCHED                 0

ENTER REFRESH PF1 HELP PF3 PREV PAGE CLR MAIN MENU
4B :02.5 01/01
Connected to host 69.24.196.48 (SC0TCP22) 11:14 PM

```

Figure 4 CPU Time Used by ICVTSD=000 and ICV=10000

One final benchmark was performed to see the effect that the default values of ICV=1000 and ICVTSD=500 versus the recommended values of ICV=10000 and ICVTSD=0. The statistics interval was set to five minutes.

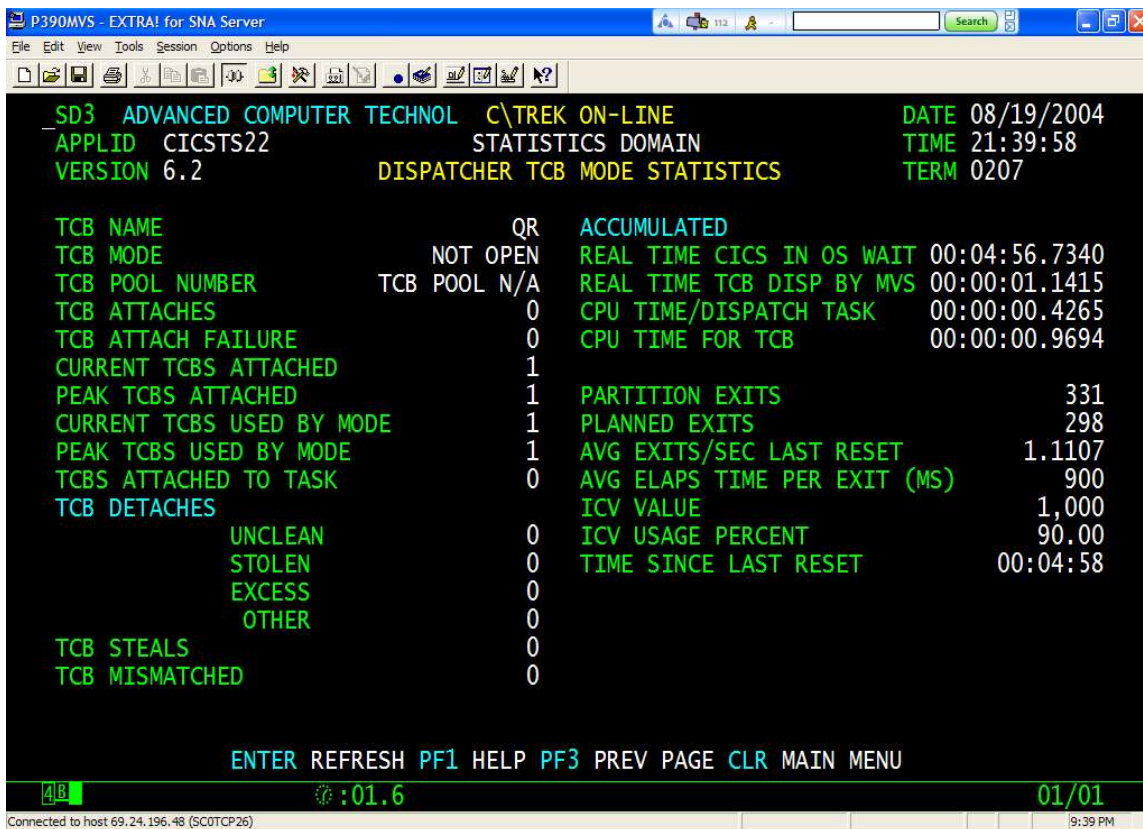


Figure 5 CPU Utilization ICVTSD=500 and ICV=1000

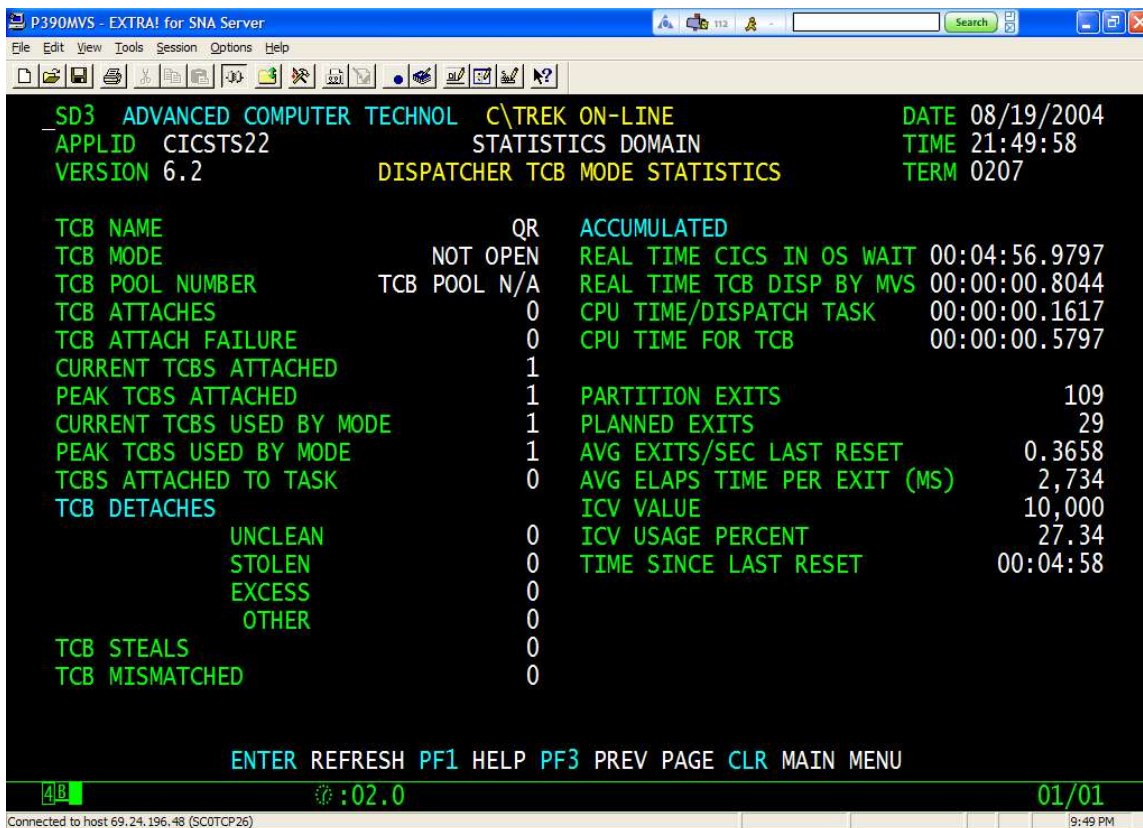


Figure 6 CPU Utilization ICVTSD=000 and ICV=10000

The results were impressive. The default values yielded a CPU utilization of .9694 seconds versus the recommended SIT values that yielded a CPU utilization of .5797 seconds. The difference was .3897 seconds or an increase of 67%. However, it appears that the default quantities for all interval control SIT values are not optimum and should be changed.

Three timer values used within CICS and defined in the SIT have been reviewed in this article. The total CPU savings will vary by installation but the more CICS regions that are executing within the image, the higher the savings that can be received when the values are changed to the values reviewed in this article. Changes to these parameters should be implemented test and quality/model office CICS regions as soon as possible.