

## Tuning Temporary Storage in CICS/TS

March 13, 2006

By: Eugene S. Hudders

### CICS/TS Performance Tuning Using C\TREK Course

This seminar covers a lot of tuning material and will require 8 hours a day. The total time for the seminar is 40 hours.

The followings topics included in this course are:

- Introduction to CICS Performance Tuning
- Using Operating System Information to Tune CICS/TS
- Tuning CICS/TS Processor Cycles
- Tuning Real Storage in CICS/TS
- Tuning Virtual Storage in CICS/TS
- Tuning CICS/TS Transaction Controls
- Tuning On Line VSAM Files
- Tuning NSR Files in CICS/TS
- Tuning CICS/TS LSR Buffer
- Tuning the Index CISZ
- Reviewing VSAM Free Space
- Reviewing the DB2 Interface

If you wish to attend this course, please send us an email at [ctrek@actpr.com](mailto:ctrek@actpr.com)

Tuning Temporary Storage (TS) has become an important item in CICS systems. TS is an area that requires attention in regions where TS is heavily used. TS come in two flavors, TS MAIN where the information is maintained in the ECDSA (Extended CICS Dynamic Storage Area) and TS AUX where the information is maintained in a VSAM file called DFHTEMP. A major factor with TS use is that more applications are beginning to depend on TS and use it with permanent information that is needed across CICS restarts. Somewhere along the line, the concept of “Temporary Storage” was lost because it is no longer considered temporary and has become “Permanent Storage” in nature. The process to tune TS MAIN is different and less complex than tuning TS AUX. A question that is often asked is: With all the storage available above the line, why not convert all TS AUX to TS MAIN? This is a good question and we will address it in this article along with other tuning recommendations.

Lets us first address TS MAIN that allows us to store the necessary queue information in virtual storage. Access to the queues is fast as all that is required is a table look-up using a Digital Tree Node (DTN) to locate the data in the ECDSA. Therefore, the first consideration when using TS MAIN is to ensure that there is sufficient virtual storage available above the line to reduce the possibility of going Short on Storage (SOS) within the CICS region. The amount of Extended DSA (EDSA) storage available above the line is controlled by a System Initialization Table (SIT) parameter called EDSALIM. Code an EDSALIM value sufficiently large to accommodate peak virtual storage use plus a growth factor (Figure 1). This figure demonstrates that we had an EDSALIM of around 409 MB and we currently have 289 MB remaining (see red arrow).

Associated with the use of the SIT parameter EDSALIM is the Job Control Language (JCL) parameter called REGION provided in either the JOB or EXEC control statements. The REGION parameter is used to control the amount of virtual storage available (for GETMAIN purposes) to the Region or address space. As TS MAIN comes out of storage above the line, we will limit the review to storage above the line. Normally, you should try to REGION code the

size as 0M to provide the maximum address space possible of 2 GB less what is required for the operating system common areas such as the (E) LPAs, (E) CSAs, (E) SQAs and Supervisor Nucleus. As can be seen in **Figure 1**, the region size (after the allocating the operating system common areas) for this address space is 1.956 GB (see orange arrow). There is 1.515 GB (see blue arrow) available after CICS is operational. The 1.515 GB total is important because this is the amount of space remaining in the region to allocate more EDSALIM or increasing VSAM buffers (LSR and/or NSR), among other things.

STOR ADVANCED COMPUTER TECHNOL		C\TREK ONLINE		DATE				
APPLID CICSTS23		C\TREK SUMMARY SCREENS		TIME 16:55:36				
VERSION 6.3		CICS STORAGE ANALYSIS		TERM 0206				
	UDSA	CDSA	SDSA	RDSA	ECDSA	EUDSA	ESDSA	ERDSA
SIZE (K)	1,024	512	256	256	63,488	34,816	2,048	19,456
AVAILABLE (%)	100.0	17.9	96.8	10.9	29.4	50.0	45.9	3.4
IN USE (%)	.0	82.0	3.1	89.0	70.5	50.0	54.1	96.5
PAGE SIZE	4096	4096	4096	4096	4096	1048576	4096	4096
EXTENTS	1	2	1	1	13	4	2	12
STORG VIOL	0	0	0	0	0	0	0	0
GETMAINS	2	4	0	0	44	52	0	0
FREEMAINS	2	4	0	0	54	69	0	0
CUSHION RELSE	0	0	0	0	0	0	0	0
TIMES AT SOS	0	0	0	0	0	0	0	0
SHORT ON STRG	NO	NO	NO	NO	NO	NO	NO	NO
						BELOW		ABOVE
STORG PROTECT	YES	REGION SIZE	(K)	9,192			1,956,864	
COMND PROTECT	YES	AVAIL BYTES REGION	(K)	2,532			1,515,472	
TRANS ISOLATION	YES	(E) DSA LIMIT	(K)	6,144			409,600	
STORAGE RECOVERY	YES	(E)DSALIM FREE BYTES	(K)	4,096			289,792	
RE-ENTR PROGS	PROT	LSQA/SWA STRGE ALLOC	(K)	296			13,024	
		USER STORGE ALLOC	(K)	6,364			428,368	
CURSOR+ENTER IN DSA - PPA INFO PF4 - MORE INFO								

Figure 1. Virtual Storage Availability

Coding a region size of 0M for CICS may not be easy and may receive some resistance from the z/OS system programmers who may have been burnt by allowing this parameter to be used for programs that will allocate as much virtual storage as available from the REGION parameter. Examples of this type of programs are compilers/assemblers and sort programs. CICS is not the type of program that dynamically expands to use all the virtual storage allocated in the REGION parameter. CICS allocates only what it needs to execute including the EDSALIM although dynamically CICS may expand due to the addition of resources or the dynamic increase in the EDSALIM size via a CEMT command. The capacity to be able to expand manually the EDSALIM size is one major reason to code a very large region size such as 0M to be able to cope with unexpected virtual storage demands that may occur with the use of TS MAIN among other things such as increased transaction volume. One final word on this topic is that your installation may have an IEFUSI SMF exit active that controls the size of allocated virtual storage. So, you may not get the amount of virtual storage requested on the REGION parameter because the IEFUSI exit may be in control of the virtual storage allocation in your system. You would need to talk to your z/OS system programmer about this exit.

So, the trade off for using TS MAIN is that you can access the data quickly without an I/O operation but you may expose the system to SOS conditions or page faults (paging) due to the increased use of real storage that is required to back the virtual storage used by TS MAIN. The SOS possibility is particularly true in installations that have many “orphaned” TS queues or queues that are created but not deleted when their need has expired (Figure 2). This display identifies those queues that have been in the system and have been unreferenced for a period of time. Although the same can be said about running out of TS space on DFHTEMP, an SOS condition probably has a greater negative effect on CICS because you may not be able to take corrective action (e.g., delete tasks and queues) and the only alternative requires the re-cycling of the CICS region. An SOS condition has a more paralyzing effect on CICS than a full DFHTEMP. Installations that have been bitten by many “orphaned” queues usually have either a monitor program that periodically deletes old unreferenced queues or have a manual on-line transaction or performance monitor that can be used to delete identified queues.

```

TSCR ADVANCED COMPUTER TECHNOL C\TREK ON-LINE 0A4620C0 DATE 09/20/2004
APPLID CICSTS23 TEMPORARY STORAGE DOMAIN TIME 20:31:24
VERSION 6.3 OLDEST REFERENCED TSQUEUES TERM 0206

D |-- QUEUE ID --| DATE TIME DATE LAST TIME LAST TIME ELAPSED
  GENE 09/20/2004 16:53:02.2 09/20/2004 18:46:37.8 000:01:44:46.2
  GREG 09/20/2004 16:53:45.7 09/20/2004 18:47:30.2 000:01:43:53.8
  MANNY 09/20/2004 16:54:40.3 09/20/2004 18:48:03.7 000:01:43:20.3
  JAMIE 09/20/2004 16:54:20.6 09/20/2004 18:48:20.6 000:01:43:03.4
  JOHANNA 09/20/2004 16:55:02.7 09/20/2004 18:48:38.7 000:01:42:45.3
  JOHN 09/20/2004 16:53:29.6 09/20/2004 18:49:42.0 000:01:41:42.0
  CTREKHL 09/18/2004 14:16:06.7 09/20/2004 19:24:01.1 000:01:07:22.9

ENTER REFRESH ENTER+CSR DETAIL PF1 HELP PF2 SEARCH PF3 PREV PAGE
PF5 MEMO PF6 REFRNCD/CREATED PF7 BACK PF8 FORW PF12+CSR DELETE TS CLR MENU
  
```

Figure 2. C\TREK Orphan Queue Display

In addition to the SOS exposure to using excessive TS MAIN, another potential problem lies in that TS MAIN queues not only require virtual storage but also require real storage when referenced by the program. So, getting better response to TS requests by using TS MAIN may be offset by paging operations that may result when there is insufficient real storage in the system to comply with the workload or when older queues are referenced for which the operating system has “stolen” the real storage due to long unreferenced periods of time. A page fault in CICS is worst than having to do an I/O operation to DFHTEMP even though both are I/O operations. The difference is that task waiting to write a record to DFHTEMP is placed

to “sleep” (wait) by CICS and allows CICS to dispatch any other task that is ready to run. So, CICS continues to operate while the task waits for the I/O completion. However, in the case of a page fault, the entire CICS address space is put to “sleep” (wait) without CICS being able to dispatch another task. So, the use of TS MAIN has to be concerned with the amount of real storage available in the system and the current system paging (Figure 3). Of particular interest on this figure are two fields: (1) The Average System Unreferenced Interval Count (UIC) that provides a quick overview as to how much real storage stress is occurring in the system. A value of 2540 indicates little stress while a value below 1000 shows a lot more stress and (2) the paging rate that indicates the amount of paging occurring in the system at this time. z/OS is designed to support high levels of paging depending on the auxiliary paging subsystem configuration. The previous data can be used to determine how the overall system is managing real storage. Modern computers have an access to large real storage resources but you must consider that a particular CICS is just one of the real storage users in the system. The paging subsystem should be analyzed to ensure that there is little or preferably no paging before expanding the use of TS MAIN in CICS/TS.

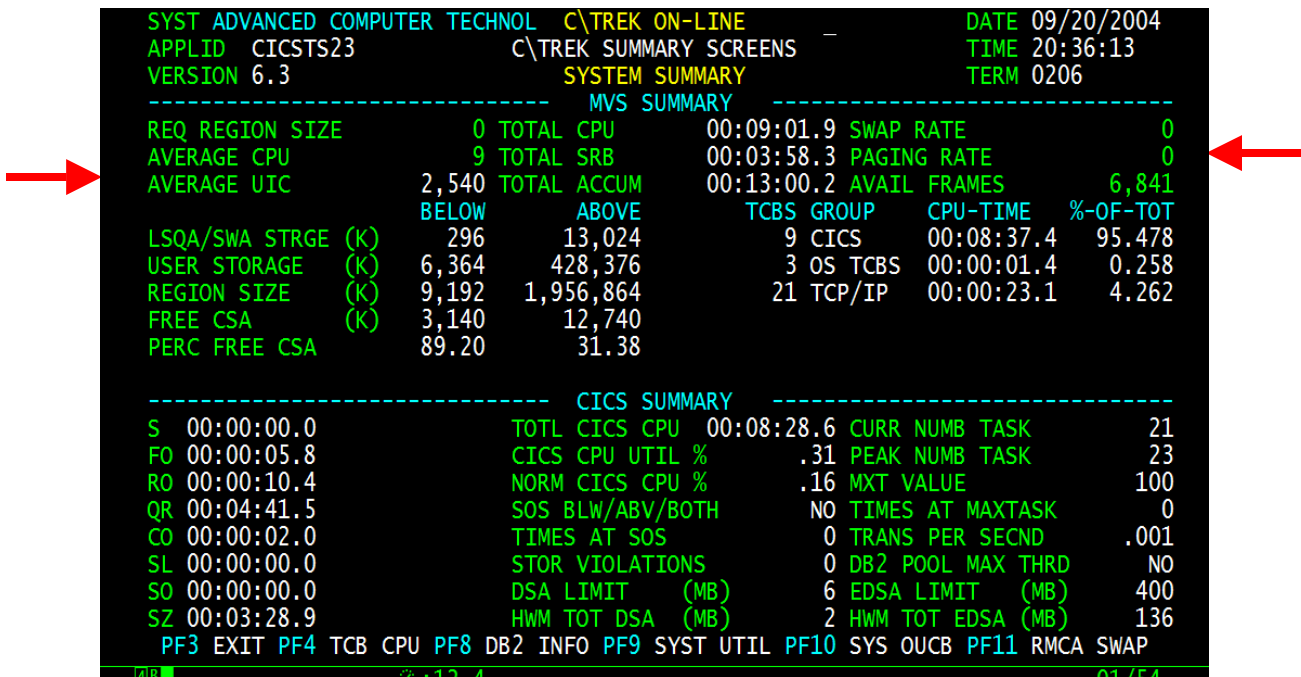


Figure 3. Overall System Display

Many installations have looked at eliminating the DFHTEMP data set and using TS MAIN for the queues. The basis for this decision is that there is sufficient virtual storage available above the line to replace the DFHTEMP data set. The calculation of how much additional virtual storage has to be added to the EDSALIM is simple. You must determine the number of CIs available in the DFHTEMP data set and multiply it by the CISZ. For example, if the DFHTEMP data set has 18,000 (100 cylinders) with a CISZ of 4 KB, then the total number of bytes handled by the data

set is 73,728,000 or 72 MB. So, if DFHTEMP is to be converted to TS MAIN, then the EDSALIM has to be raised by at least 72 MB.

You probably would want to raise the EDSALIM by a larger amount because DFHTEMP could be specified with a secondary allocation amount that would only be used when the primary allocation is full or to allow for future growth. Some installations use the peak number of CIs used for this calculation instead of the CIs allocated plus a buffer amount (“fudge factor”). This option is probably valid in a static environment with little growth or changes. However, if this area is not constantly monitored, you could wind up with SOS conditions due to application growth or new applications being implemented in the CICS. Using the peak use to base the required EDSALIM increment could be short sighted and probably would require continuous monitoring. This takes time, which is something many system programmers do not have.

The maximum number of CIs that can be defined for DFHTEMP is 65,535 (X'FFFF'). The maximum CISZ is 32,768. So, the maximum TS size that could be handled by TS AUX would be  $((65,535 - 1) * 32,768)$  or almost 2 GB. There is a header record in DFHTEMP that is not used for data thus the total number of CIs has to be reduced by one. Therefore, eliminating TS AUX would depend on the total allocated virtual storage that would be available in the region after all the CICS requirements (control blocks, VSAM buffers, EDSALIM, trace table, etc.) were met. This total may vary from 1.2 to 1.7 GB in the majority of installations. However, remember that virtual storage eventually has to be backed by real storage. Lack of virtual storage to handle TS MAIN will result in SOS conditions within CICS. Finally, there is one other factor that should be considered when considering the change of TS AUX for TS MAIN. DFHTEMP tends to be self-adjusting to unused or unreferenced queues, that is, older queues will eventually migrate to DFHTEMP and not remain in the TS buffers. In the case of unused TS MAIN queues, they will still be in virtual storage but should not require any real storage. The solution of assigning all of TS to MAIN should only be applied to high performance and high use TS regions. There is another alternative with DFHTEMP that may yield good results without having the exposure of SOS and investing a lot of virtual and real storage. This option is reviewed later in this article when we discuss buffering for DFHTEMP.

Finally, if the decision is made to force all TS queues to TS MAIN, an increase in the EDSALIM is probably required. Therefore, you must ensure that by allocating additional virtual storage to the EDSALIM you do not reduce the amount of virtual storage available in the region for other tuning tasks such as additional LSR buffers or future growth. For example, imagine that your current system has 1.5 GB available after CICS is loaded and executing. You now want to replace DFHTEMP and send all the queues to TS MAIN. Imagine that the analysis of how much space is required to place all TS into MAIN yields a requirement of an additional 1.3 GB increase to the EDSALIM to accommodate the space required by DFHTEMP. If 1.3 GB were allocated, then the entire region would only have 200 MB left for growth in other areas. This may be a concern for future expansion, specifically LSR/NSR buffer tuning or if you are interested in increasing the MXT value.

The quoted general “rule of thumb” (ROT) for the use of TS MAIN is for queues that are small in size, have a short life span and require fast access. However, what is small in size, what is a short life span and what is fast access? When the amount of virtual storage available on a system was 16 MB or the real storage available on a system was 256 KB (I know, some of you may have never seen a machine with this amount of real storage), small probably meant less than 1 KB. However, now that many systems have 2 GB or more of real and virtual storage and CICS can use above the line storage, the word “small” is misleading and can be interpreted to be a larger size such as 4 or 8 KB or even larger. The same concerns apply to “short life span and fast access”. Therefore, the previous ROT may be obsolete although it still does provide a guideline of when to use TS MAIN. In addition, today’s application programmers may not be as concerned with efficiency because of the availability of faster, higher capacity mainframes and disk subsystems to follow any ROTs. This is especially true if the simple task of TS queue clean up by the application is ignored. However, there should be some guideline established or excessive use of TS MAIN across many CICS systems may cause the overall system to page affecting the overall response times and performance of every address space in the system.

So new guidelines for TS MAIN usage may be required that fit the current processing environment. In CICS systems that have very little TS activity, then forcing all TS queues to TS MAIN could improve performance. This is accomplished by setting the number of buffers and strings on the SIT TS= parameter to zero. Another method is to define a series of TS models that could be used to force the TS queues to MAIN. In systems where there is a significant amount of TS use, then other factors must be analyzed. As previously mentioned the use of TS MAIN has to be balanced against the amount of virtual storage available to CICS both at the region and extended DSA levels and the overall paging in the system. If there is an insufficient allocation of virtual storage, excessive use of TS MAIN may be a liability. Also, if the system is already paging, increased use of TS MAIN could worsen the condition and affect overall system performance. This is especially true if there are many CICS regions executing in the system and the decision to increase TS MAIN usage is made across the board to all CICS regions.

Therefore, a new ROT for TS MAIN could be based on the size of the record to be written but not necessarily as the prime guideline. The prime guidelines would be little or no paging activity and availability of sufficient virtual storage in the EDSALIM. Another guideline could be established for data that are important to the application (versus transaction). In this case the TS queue should be kept in TS MAIN for fast access. Highly volatile TS data, that is, data that is created, read and deleted in a short span of time would also be suited for TS MAIN. This is especially true for volatile TS queues that are comprised of many small records. These TS queues should always be written to TS MAIN to reduce the overhead of using DFHTEMP caused by buffer compressions when the queue is deleted. Small in this case refers to the TS segment size. A TS segment size depends on the CI size requested for the DFHTEMP data set. For a DFHTEMP CISZ greater than 16 KB, the segment size is 128 bytes long while the segment size will be 64 bytes for a DFHTEMP CISZ of 16 KB or less. Finally, TS MAIN could

be used for single record queues versus multi-record queues that should probably go to DFHTEMP.

There many considerations when tuning DFHTEMP used for TS AUX. The primary TS performance consideration is the number of buffers allocated. The unfortunate fact is that the default number of buffers (three) is probably not good for installations that use TS AUX. Note that the number of strings plays a secondary role in this tuning process and will be covered later in this article. TS I/O operations are deferred for non-recoverable queues. That is, the buffer is not written to the disk until the buffer is required for another request. Therefore, if sufficient buffers are allocated, the number of physical I/O operations to the disk file may significantly reduced or eliminated because the requested TS data can be found in the buffer. Thus, if sufficient TS buffers are available, then the buffer pool serves as a “simulated TS MAIN” without having to sacrifice a lot of virtual storage and/or face the exposure of SOS conditions. Naturally, even if the record is found in the buffer, it is slower than TS MAIN but it is still faster than accessing the disk. Recoverable TS queues are written immediately to disk.

Unfortunately, there isn't a formula available to determine the number of buffers to allocate to DFHTEMP to get a good hit ratio because a lot depends on when the queues are accessed. So, in many cases, determining the number of buffers required is accomplished by trial and error. In general, you would like to achieve an 80% or higher hit ratio or a low physical I/O to logical GETQ/PUTQ AUX such as 10 to 20%. Therefore, you would have to monitor and add buffers as required. However, if a good hit ratio can be achieved, the total amount of storage required will be much less than reserving the entire DFHTEMP space in virtual storage for TS MAIN use.

Let us review a user case study that illustrated this point. The user had a 100 cylinder TS AUX data set using a 4K CISZ into virtual storage. In this case, DFHTEMP had 18,000 CIs. Eliminating DFHTEMP and converting the TS AUX to TS MAIN required an EDSALIM increase of 72 MB ( $4096 * 18000$ ). However, after some trial and error we found that assigning 500 buffers to DFHTEMP we could achieve a hit ratio of over 80%. The virtual storage requirement for the 500 buffers was 2 MB ( $500 * 4096$ ). Additional buffers could improve the hit ratio but there is a point where the additional investment was not worth the return. Naturally, as with all benchmarks, your mileage may vary and you will have to experiment. Each TS region is different and may require different investments to achieve the desired hit ratio.

The advantages of using additional TS buffers are that you can improve the response time with a minor investment of real and virtual storage. In addition, as non-recoverable queues are not always written to disk, the amount of used disk space in DFHTEMP is less. The disk space is actually used by little referenced or unreferenced queues that do not occupy space in virtual storage within CICS. The major disadvantage is that even with a good hit ratio of 80% or better, TS AUX is not as fast as TS MAIN. This is the trade off that must be decided by each installation. The buffer hit ratio is measured by the following formula:

$$\% \text{ Hit Ratio} = (100 - ((\text{Buffer Writes} + \text{Buffer Reads}) / (\# \text{ PUTQ AUX} + \# \text{ GETQ AUX}) * 100))$$

Let us review the need for TS AUX strings. The default number of strings in the SIT is three. The default number of strings may be insufficient depending on the number of buffers allocated and TS activity. In many installations, TS strings are over allocated. Tuning short on TS string conditions should start by tuning the number of buffers allocated and not by simply increasing the number of strings available. A string is needed when a physical I/O is required to DFHTEMP. So, if we manage to reduce the number of I/O operations by 80% (desired TS hit ratio), then the activity requiring strings is significantly reduced. So, to correct a short on TS strings condition, first ensure that we have the proper number of TS buffers and that we are achieving the desired hit ratio. The number of recoverable TS requests will affect the physical I/O activity. So, in general you would like the peak number of strings used to be no higher than 70% of the strings allocated. Increase the number of strings allocated if you are achieving your TS hit ratio objective and you are still getting short on TS string conditions. Even if your strings use exceeds 70%, you should not increase the figure unless you have short on string conditions and your buffer hit ratio is 80% or better.

Another area that requires attention is writes greater than TS AUX CISZ. Temporary Storage permits this condition to exist and handles it more efficiently in CICS/TS than in previous CICS versions. This feature allows the user to define a CISZ that accommodates the majority of the TS queues while handling exception queues that have records that exceed the TS AUX CISZ. This reduces the need to have a very large CISZ defined to handle a small number of TS records and thus, save on virtual storage. In other words, some writes greater than the CISZ are not always bad. Although TS handles the situation, performance is affected by this condition in that there is extra overhead to process the longer record and more than one buffer is required to read or write the TS data to DFHTEMP. In the case where there are insufficient buffers, the read of a CI that is longer than the CISZ may require multiple I/O operations before the entire CI is in storage and ready for processing. In other words, multiple buffers are required to process records that are greater than the DFHTEMP CISZ. The use of multiple buffers may affect the overall number of buffers available for the look aside hit ratio previously discussed.

CICS identifies the condition by two statistics. The first statistic indicates how many writes were greater than the defined TS AUX CISZ. The objective is to keep this value below 3% of the total number of PUTQ AUX. The second statistic indicates the longest record written to DFHTEMP. This is a High Water Mark (HWM) statistic and should not be used to determine the correct CISZ for DFHTEMP. The reason that this statistic is not good for this purpose is that it does not reflect how many times this condition occurred. For example, you may have a CISZ that is 4 KB in size. CICS may provide a statistic that indicated that you had "n" writes greater than the CISZ

and that the longest record was 28 KB in length. If you select a CISZ of 28 KB based on this statistic, you may be wasting storage because you may have had the condition where you wrote one record of 28 KB and “n-1” records that were 6 KB or less. So, setting the CISZ to 6 KB would eliminate all but one of the writes greater than the CISZ. Unfortunately, CICS does not provide this distribution. Correcting this condition may require several trials before the proper size is found. The best method is to select a CISZ that accommodates 75% of the TS queues that exceeded the TS AUX CISZ. This requires analyzing the TS queues in the system (Figure 4 Parts 1 and 2). The first thing to determine is what was the HWM longest record written to DFHTEMP (Figure 5). In this case, the longest record written was 28,000 bytes. Using Figure 4 (Parts 1 and 2) we can determine that we can cover 75% or more of the queues when we have a 10 KB CISZ.

```

TSRC ADVANCED COMPUTER TECHNOL  C\TREK ON-LINE          DATE 09/20/2004
APPLID CICSTS23                  TEMPORARY STORAGE DOMAIN  TIME 21:17:38
VERSION 6.3                      RECOMMENDATIONS          TERM 0206

PUT/PUTQ AUX          19 BUFFER READS              0 PEAK CIS USED          85
GET/GETQ AUX          0 BUFFER WRITES             34 % CIs USED (PEAK)    23.68
WRITE > CISIZE        18 CI SIZE              4,096 CURR CIS          85
NUM TS STRINGS        3 CIS ALLOCATED          359 % CURR CIs         23.68
NUM TS BUFERS         50 PUTS PER SECN         0 # AVL BYTE/CI        4,032
I/O PER SECN         0 GETS PER SECN          0

CISZ  QUEUES  PERC  ACCUM  RECOMMENDED CISZ  10,240
512   0       0.00  0.00  CURRENT CYL ALLOCATED  2
1,024 0       0.00  0.00  RECOMM CYL WITH NEW CISZ  5
1,536 0       0.00  0.00
2,048 0       0.00  0.00  RECOMMENDED NUM OF BUFFERS  50
2,560 0       0.00  0.00  RECOMMENDED NUM OF STRINGS  3
3,072 0       0.00  0.00
3,584 0       0.00  0.00
4,096 0       0.00  0.00  EDSALIM REQUIRED FOR
4,608 1      16.60  16.60  TS AUX TO TS MAIN (MB)
5,120 0       0.00  16.60  MAX ALLOC  1.40
5,632 0       0.00  16.60  PEAK USED  0.33
6,144 2      33.30  49.90
ENTER REFRESH PF1 HELP PF3 PREV PF7 BACK PF8 FORW CLR MAIN MENU  MORE D
4B  :02.9 01/54
    
```

Figure 4. EDSALIM Requirements for TS MAIN (Part 1)

```

TSRC ADVANCED COMPUTER TECHNOL C\TREK ON-LINE DATE 09/20/2004
APPLID CICSTS23 TEMPORARY STORAGE DOMAIN TIME 21:19:03
VERSION 6.3 RECOMMENDATIONS TERM 0206

PUT/PUTQ AUX 19 BUFFER READS 0 PEAK CIS USED 85
GET/GETQ AUX 0 BUFFER WRITES 34 % CIs USED (PEAK) 23.68
WRITE > CISIZE 18 CI SIZE 4,096 CURR CIS 85
NUM TS STRINGS 3 CIS ALLOCATED 359 % CURR CIs 23.68
NUM TS BUFRS 50 PUTS PER SECN 0 # AVL BYTE/CI 4,032
I/O PER SECD 0 GETS PER SECD 0
CISZ QUEUES PERC ACCUM RECOMMENDED CISZ 10,240
 6,656 0 0.00 49.90 CURRENT CYL ALLOCATED 2
 7,168 1 16.60 66.60 RECOMM CYL WITH NEW CISZ 5
 7,680 0 0.00 66.60
 8,192 0 0.00 66.60 RECOMMENDED NUM OF BUFFERS 50
10,240 1 16.60 83.30 RECOMMENDED NUM OF STRINGS 3
12,288 0 0.00 83.30
14,336 0 0.00 83.30 EDSALIM REQUIRED FOR
16,384 0 0.00 83.30 TS AUX TO TS MAIN (MB)
18,432 0 0.00 83.30 MAX ALLOC 1.40
20,480 0 0.00 83.30 PEAK USED 0.33
22,528 0 0.00 83.30
24,576 0 0.00 83.30
ENTER REFRESH PF1 HELP PF3 PREV PF7 BACK PF8 FORW CLR MAIN MENU MORE B
  
```



Figure 4. EDSALIM Requirements for TS MAIN (Part 2)

```

TSSU ADVANCED COMPUTER TECHNOL C\TREK ON-LINE DATE 09/20/2004
APPLID CICSTS23 SUMMARY SCREENS TIME 21:53:55
VERSION 6.3 TEMPORARY STORAGE SUMMARY TERM 0206

PUT/PUTQ MAIN 0 NO. TS BUFRS 50 CI SIZE 4,096
GET/GETQ MAIN 20 NO. BUFR WAIT 0 CIS ALLOCATED 359
PUT/PUTQ AUX 19 USR W/ BUFRS 0 PEAK CIS USED 85
GET/GETQ AUX 0 PEAK W/ BUFR 0 % CIs USED (PEAK) 23.67
FORMAT WRITES 0 BUFFER WRITES 34 CURR CIS 85
WRITE > CISIZE 18 BUFFER READS 0 # AVL BYTE/CI 4,032
FORCED WRITES 0 MAX WRTE BUF 25 MAX CIS FORMAT 359
I/O ERRORS 0 CUR WRTE BUF 25 SEGM PER CI 63
TS COMPRESS 0 BYTES PER SEGM 64
ENTR LNGST Q 15 BUFFER LOCKS 0
LNGST AUX RLEN 28,000 % BUFR READS .00 STRING LOCKS 0
TSNAME IN USE 7 % BUFR WRITES 178.94 # TS STRINGS 3
PEAK TS NAMES 7 % TS BUFR I/O 178.94 PEAK STRINGS 1
TIMES Q CREAT 0 RECOM BUFFERS 200 STRING WAITS 0
                                PEAK WAIT STR 0
                                CURR STOR TS 24 % STRINGS USED 33.33
                                PEAK STORAGE 24
                                STORG EXHAUST 0
                                USR W/ STRG 0
ENTER REFRESH PF1 HELP PF3 PREV PAGE PF4 TS QUEUES CLR MAIN MENU
  
```



Figure 5. Temporary Storage Statistics

Changing the CISZ brings us to another area that requires attention, which is the number of CIs allocated to DFHTEMP and the current/peak number of CIs used. If the entire DFHTEMP file is used and we increase the CISZ and do not increase the space allocation, then we run the risk of being short on TS AUX storage that affects response time and could have a negative impact on system reliability. One of the first areas that have to be analyzed is that there is a minimum of orphaned queues, that is, that there are a minimum number of unreferenced queues that have been in DFHTEMP for a long period of time. The solution to this situation is to ensure that the application programmers perform proper queue clean up when finished using the queue. Incorrect clean up usually results from error or cancellation conditions not properly handled by the application program code. Some installations have found it easier to write clean up programs that delete unreferenced queues after a certain amount of aging than to get the application programmers to clean up the program code. Other installations have fixed the problem by making DFHTEMP very large or by defining secondary extents to handle unexpected increased usage.

If the CISZ is increased, then consideration should be given to increasing the allocated disk space because if the same allocated space is used, then the total number of CIs allocated will be less than previously defined. For example, taking the situation where there are 100 cylinders of a 4 KB CISZ for DFHTEMP that resulted in having 18,000 CIs. However, if we increase the CISZ to 8 KB, then we would only have 9,000 CIs in DFHTEMP. This reduction in the total number of CIs allocated could result in the peak number of CIs used to be a higher percentage than when the CISZ was smaller. Therefore, whenever increasing the DFHTEMP CISZ, you should also consider increasing the total space allocated to maintain the same number of CIs as with the previous CISZ. The physical use of the CIs in DFHTEMP will be dependant on how well you have buffered TS AUX. If you receive a high hit ratio, then there will be less physical activity on DFHTEMP and the peak CI use will be lower. If the TS AUX buffering is not adequate, then the CI use on DFHTEMP will be higher.

The objective is to maintain the peak number of CIs used to less than 70% of the total CIs allocated on DFHTEMP. The extra 30% serves as a buffer to avoid the space exhaustion of DFHTEMP that results in tasks waiting for TS space or for a secondary space to be allocated and formatted. The DFHTEMP out of space wait could cause the system to stall and require a recycling if not corrected on a timely basis. CICS provides the statistics regarding the total number of CIs allocated, the number of CIs currently in use and the peak number of CIs used. The percent of use is not provided by CICS and has to be computed.

As a protection against running out of TS AUX space, we recommend that the DFHTEMP cluster definition include a secondary allocation on the space definition. If there is insufficient space on DFHTEMP and a secondary allocation can be made, CICS will format another extent up to the maximum number of extents allowed by VSAM or maximum number of CIs supported by DFHTEMP. However, formatting secondary extents is a costly operation and should be used only as a safety valve. The primary allocation should be large enough to accommodate the

peak number of CIs while leaving around 30% for growth. CICS provides a statistic that can be used to identify when a secondary extent has been allocated. This statistic is called “format writes” and should be zero (Figure 5). If a secondary extent is allocated (i.e., format writes > 0), then the DFHTEMP file should be redefined with a larger extent to account for the new size. Finally, if you are going to pay the price of a secondary extent, make sure that the secondary value is sufficiently large so as to avoid having to re-allocate a new extent because the previous one filled up. In other words, don’t request small secondary allocations such as one cylinder.

Two other TS AUX areas that should be analyzed are I/O errors and TS compressions. There should never be an I/O error on DFHTEMP because of modern disk technology. Should an I/O error occur, redefine DFHTEMP on another volume. TS compressions occur as TS queues are deleted within a CI because the deleted space is spread across the CI and may not be contiguous. So, CICS compresses the queues to the front of the CI leaving the contiguous free space at the end for future use. TS compressions take CPU time to perform. You may see a certain amount of TS compressions occur in a system that has a good look aside hit ratio because the CIs in buffers are reused in virtual storage. This occurs even though the CI may not have been written to DFHTEMP. Although compressions are an overhead, the fact that there was no I/O operation is better. There is very little that can be done to control TS compressions when using good buffering techniques. One possible action is to ensure that highly volatile queues, especially those containing small records, are written to TS MAIN.

Tuning TS depends on whether the TS queue resides in TS MAIN or TS AUX. The areas to tune vary depending on whether the TS queue is in virtual storage or disk. This article reviewed areas that should be reviewed when tuning Temporary Storage.

If you have any questions, comments or request for more information, please contact us:

Address: C\TREK Corporation PO BOX 560069 MONTVERDE FL 34756	Phone: (407) 469-3600 (787) 756-5620	C\TREK Corporation Advanced Computer Technology
E-mail: <a href="mailto:ctrek@actpr.com">ctrek@actpr.com</a>	Fax: (787) 756-5150	
Website: <a href="http://www.ctrekcorp.com">www.ctrekcorp.com</a>	Support: (787) 397-4150 (321) 297-5838 (787) 462-0406	