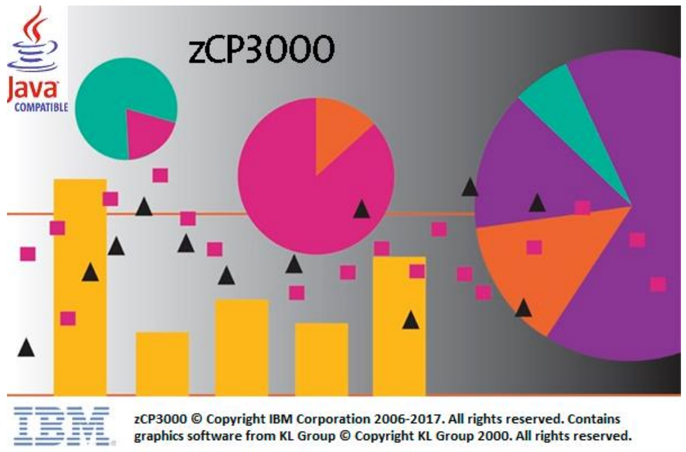
IBM Washington Systems Center

CP3KVMXT – data extract for zCP3000 and zPCR User's Guide for Data Collection


zCP3000
Java COMPATIBLE
IBM
zCP3000 © Copyright IBM Corporation 2006-2017. All rights reserved. Contains graphics software from KL Group © Copyright KL Group 2000. All rights reserved.

CP3KVMXT Version 2.8b

March 2018

Level1 support : cpstools@us.ibm.com

Author/level2 : Gretchen Frye frye@us.ibm.com

Table of Contents

Objectives

The CP3KVMXT utility program reads CP monitor data in order to produce a file which is directly useable by zCP3000 and zPCR. It is intended for use by customers to extract performance information which will then be used by IBM employees or IBM Business Partners to produce a capacity plan.

CP3KVMXT and its prerequisite packages, Monview and VMARC, may be obtained from the IBM VM Download Packages website at http://www.vm.ibm.com/download/packages/. Use of tools from this site is conditional on the “Agreement for Downloading from the IBM VM Download Library” at http://www.vm.ibm.com/download/license.html Please note that the zCP3000 modeling tool itself is available only to IBM employees and entitled IBM Business Partners.

IBM employees and IBM Business partners can access additional tools for capacity planning, including zCP3000, zPCR, and the CP2KEXTR z/OS data extract facility, can be found on the Capacity Planning Support website at

http://w3.ibm.com/support/americas/wsc/cpsproducts.html

or on the Business Partners website at http://partners.boulder.ibm.com/src/atsmastr.nsf/WebIndex/PRS1762 .

Changes

Please note that the cp3kvmxt code will be updated more frequently than the user's guide. Code releases with only bug fixes and minor functional updates will not have a corresponding user's guide update.

March 2018 – New PRBT CPUMF metric for problem state time; new field SVM identifies Service Virtual Machines; new fields SPX, SPXDA, and WSPXDA indicate if VM65396 is installed and active.

November 2017 – New SMTEFFTD thread density field (GCP and IFL). New absolute capping field (all engine types, all partitions). See Appendix E "Table 3: SYS information about this z/VM image and LPAR information" for field definitions. More help was added to the section "What if you collect hourly monitor data files?" New planning help for guest operating systems in the section "If you are running guest SCPs with lots of defined i/o".

August 2017 – support for z14 CPUMF data; new FILELIST option to allow multiple input mondata files; support concatenated mondata input; bug fix when real engines were vary off/on during measurement data. See CP3KVMXT Options for more about FILELIST. See Appendix A: Syntax for more about how to use this option.

November 2016 – new fields TV, WVTM, WVTML for sys and user T/V ratio; new fields SSINM and SSINAMES (reserved for future use); filter out unused monitor data earlier in the process; gmt offset sometimes wrong in the preview.exec

October 2016 – Data collection problems

November 2015 – new in doc only (already in EDF), user fields relating to use of virtual disk in memory: VDISK and VDIO.

September 2015 – new TLB fields, CPUMF reqt's, nameable PARMS file; how and why to use an EDF in zPCR.

July 2015 – new SMT fields (z13 only)

October 2014 – "Recovering from Problems" in the Saved Monitor Data cookbook.

July 2013[1] – New system memory fields PGES and PAGESLOT. Updated Problems appendix, updated monitor disk space estimate.

[1] Editor's note: excluding for brevity changes older than July 2013. Please contact the author if you need something from a more ancient version of this doc.

Overview

CP3KVMXT : VM Extract Utility

The zCP3000 VM extract utility (CP3KVMXT) generates an Enterprise Data File (EDF) describing the machine, LPAR config, and details for this z/VM system, for input into the zCP3000 and zPCR tools (see below) from one or more monitor data files. The fields and macros generated by CP3KVMXT are documented in an appendix of this document.

zCP3000 Performance Analysis and Capacity Planning Tool

zCP3000 is available to IBM teams and IBM Business Partners, to do performance analysis and capacity planning for installations running one or more z/OS, z/VM, z/VSE, or Linux on z partitions on one or more IBM z Systems processors. zCP3000 uses the LSPR processor performance tables in the zPCR tool, and is designed to be able to model complex configurations, including multiple partitions, multiple operating systems, partition weights and capping, hiperdispatch, SMT (simultaneous multi-threading), parked engines (vertical polarization), specialty engine usage, and the cooperative processing environment known as Sysplex.

zPCR Processor Comparison Reference

LPAR configuration information, CPUMF workload characteristics (RNI), and SMT information can be read from EDFs created on z/VM (by cp3kvmxt) or from z/OS (by cp3kextr). zPCR is designed to provide capacity planning insight for IBM z Systems platforms running various SCP/workload environments using various LPAR partition configurations. Capacity results are based primarily on IBM's published LSPR data for IBM z Systems families.

IBM CPSTOOLS

The Capacity Planning Support (CPS) team at the Washington Systems Center provides a comprehensive suite of tools designed to assist in capacity planning for IBM z Systems, including zPCR, zCP3000, zBNA, zSCON, and zSoftcap. For more information, please see the CPSTools Overview
ftp://w3-public.dhe.ibm.com/support/wsc/zMasters_2016_CPS_Tools_Overview.pdf

Or visit our website at
http://w3.ibm.com/support/americas/wsc/cpsproducts.html (IBM internal)
http://partners.boulder.ibm.com/src/atsmastr.nsf/WebIndex/PRS1762 (PartnerWorld)

Get notification of new releases by joining the zCP3000 google group at
https://groups.google.com/forum/#!forum/zcp3000

Planning For CP3KVMXT Data Collection

HMC Global Performance Data Control Authority

Is not required but is strongly recommended. If the partition has this authority, then collected monitor data will include the names, types and numbers of engines, LPAR weights, and cputime used, for other partitions on the same machine. If you are collecting data to provide the LPAR configuration information in a zPCR study, and this partition does not have Global Performance Data Control Authority, you should stop now and collect data instead from a partiton that does have authority. For zCP3000, you will also be missing valuable information about the capacity of the machine, since you will have no information about partitions sharing resource with the partition that you are interested in.

CP3KVMXT Input Data Requirements:

CP3KVMXT uses the monitor records created by the z/VM Monitor Service for input. These records must be in the format specified by the published z/VM Monitor Record Programming Interface (ie, http://www.vm.ibm.com/pubs/mon630/index.html for z/VM 6.3.0 data).

Important: CP3KVMXT will not read history or acum files, nor any sort of formatted performance report produced by the Performance Toolkit for z/VM, VMPRF, Tivoli Performance Monitor, or any OEM product. Only monitor data created by the IBM monitor service and written to disk by IBM Monwrite is supported. **Monitor data written to disk by an OEM product *may* work, but will not (necessarily) be supported.**

What's likely to cause you trouble, and what should be ok as is

Capacity planning requires data from a longer period of time than is typically used for performance tuning. Running it for one peak hour is not enough, particularly for systems running guests which may have a substantial scheduled workload that kicks off at midnight. Collecting performance data should be considered a cost of doing business. Preferably, the capacity plan will contain an entire (business) week's worth of data, prime shift or 24x7, in 15 or 30 minute "durations" (an accumulation of monitor intervals).

CP3KVMXT is compatible with realtime performance monitors such as z/VM Performance Toolkit which also use z/VM Monitor Records. Multiple users can simultaneously access the monitor service. For information on the MONDCSS saved segment and running the monitor, please refer to "Chapter 9. Monitoring Performance Using CP Monitor" in SC24-5999 z/VM Performance Guide (see Appendix B for the url of the z/VM Publications library).

Please note that the default MONDCSS size is sometimes too small, especially for guest environments when there are a lot of devices defined, and the result of this is lost data, including lost configuration records which are required by cp3kvmxt. If CP3KVMXT is using realtime data, it will shut down if the data is being dumped into the small MONDCSS at a faster rate than CP3KVMXT can process it. See the section entitled "Speeding Up CP3KVMXT", later in this document.

CP Monitor Requirements

For an excellent summary of how to collect monitor data, please refer to http://www.vm.ibm.com/perf/tips/collect.html . Use the CP Q MONITOR command to display the current settings.

CP3KVMXT requires event and sample data for the Processor, Storage, User, and I/O (dasd class) domains. If a realtime performance monitor like z/VM Performance Monitor or something similar is currently running, the monitor is already running and is probably ok like it is. Perfkit and cp3kvmxt can both read the same monitor data at the same time. The Monitor sample interval should be between 1 and 5 minutes long; real-time monitors tend to use a shorter interval, such as 1 to 2 minutes, which is fine, but there are tradeoffs when choosing the interval length:

- **1 minute interval:** this is the default, and typically what a performance monitor would want. Startup information is the same but it collects twice as many samples (device activity, user state, etc) as a 2 minute interval. This will also increase the CPU consumption and storage space of the userids running CP3KVMXT and Monwrite (because there are more samples to process).
- **2 minute interval:** This is a long time, when using a performance monitor to do some tuning. This is fine for capacity planning, except that the cp3kvmxt duration must be a multiple of 5, *AND* it must be a multiple of the monitor sampling interval, *AND* it must match the duration used in other EDFs, from this system image and any other z/VM or z/OS image that will be part of the capacity planning study. *With a 2 minute CP monitor interval, you will probably end up with a 30 minute zCP3000 duration. Keep this in mind if you are trying to collect simultaneous data from other partitions for the same study.*
- Therefore, a 2 minute interval is fine, but you will then need to use a cp3kvmxt duration of 10, 20, or 30 minutes, not a 15 minute duration. The longer the duration, the more data you need to collect, because 20 minutes of data will yield a single data point on a graph with a duration of 10 minutes (the first minute is the zero ordinal, it does not count as a sample).
- Monitor overhead can be lessened by monitoring only what you need. CP3KVMXT does not use data from the Scheduler, Seeks, or Application domains, and it uses only dasd class I/O. Here are sample commands to measure just these, sampling at 1 minute intervals:

```
CP MONITOR SAMPLE DISABLE ALL INT 1 MIN
CP MONITOR SAMPLE ENABLE PROC
CP MONITOR SAMPLE ENABLE STOR
CP MONITOR SAMPLE ENABLE USER ALL
CP MONITOR SAMPLE ENABLE I/O CLASS DASD
CP MONITOR EVENT DISABLE ALL
CP MONITOR EVENT ENABLE PROC
CP MONITOR EVENT ENABLE STOR
CP MONITOR EVENT ENABLE USER ALL
CP MONITOR EVENT ENABLE I/O CLASS DASD
```

What if you run an IBM or OEM performance product?

This is not a problem. The monitor segments in real memory are read-only, so MONWRITE can be reading monitor data at the same as PerfKit, or any other monitor. **Important:** CP3KVMXT will *not* read history or acum files, nor any sort of formatted performance report produced by the Performance Toolkit for z/VM, VMPRF, Tivoli Performance Monitor, or any OEM product. Only monitor data created by the IBM monitor service and written to disk by IBM Monwrite is supported. **Input from monitor data written to disk by an OEM product *may* work, but will not be supported.**

What if you collect hourly monitor data files?

Because monitor data files can be quite large, some installations close the monitor data file on an hourly basis in order to reduce the storage space required (ie, the LinMon tool). This is fine, however keep in mind that:

- You can run cp3kvmxt against each, one at a time, and create an EDF from each monitor data file, then merge those into a single system image EDF using the merge tool in zCP3000. But you will probably end up losing one interval per hour in those hourly EDFs.
- **(New!)** You can process more than one monitor data file in a single cp3kvmxt run, and this will create a single EDF. This would be the “Filelist” option, which can only be specified in a cp3kvmxt runtime parms file. Please refer to the CP3KVMXT Options section in this doc.
- P.S. You probably should be using a PARMS file to specify runtime options.

Important: do not use the NOCONFIG option on the MONWRITE command, even for hourly monitor data files. An EDF cannot be created without the config records. If you use the NOCONFIG option and are also using FILELIST, then the one including the config records *must* be the first one processed. And no, you can’t use the mondata file with the config records from *last* week with this week’s mondata files.

Requirements for collecting CPUMF data

Transaction rich workloads behave differently than compute-intensive workloads, but how do you know whether a Linux guest running a webserver, application server, and maybe a database server would be considered more i/o or compute intensive? CPUMF

(CPU Measurement Facility) is the best way to characterize the workload, since it looks not only at CPU time, but also at locality of reference in memory and all of the levels of processor cache (processor local, on chip, on book/drawer). CPUMF data is part of the process domain, so it is automatically collected if

- z10 at driver 76D or higher, or any machine type newer than that
- z/VM 6.2 or higher (or z/VM 5.4 w/ apar VM64961)
- Allow the partition to collect basic and extended counters (via HMC, "Change Logical Partition Security"). Please ref IBM ITSO Redpaper "Setting Up and Using the IBM System z CPU Measurement Facility with z/OS"
- No additional monitor settings are required, CPUMF is part of processor domain sampling. By default, CPUMF collection is enabled, but it is possible to specifically *disable* just the CPUMF collection, using the Monitor Sample Disable Proc CPUMFC cmd.

Simultaneous Multi-threading (SMT)

Simultaneous Multi-threading (SMT) is supported on IBM z13 hardware and above, with z/VM 6.3 and above. SMT must be explicitly enabled by the partition. If SMT is enabled, SMT metrics will automatically be included in the z/VM monitor data and in the generated EDF.

If you are running guest SCPs with lots of defined i/o

There are different problems depending on whether your z/VM is current.

If your z/VM is still in service[2], then you should be able to generate an EDF, but you may need to allow more time for the monitor service to get started and into a steady state, before the first cp3kvmxt interval is collected. You can use the "starttime" parameter in the PARMS file to do this. If you are using a 15 minute interval, then start the monitor service 20 minutes before the first interval that you want to appear in the zCP3000 capacity planning model.

Before z/VM 6.2, then it is quite likely that you will not be able to generate an EDF with the i/o enabled. The default MONDCSS and monitor settings can support only about 3600 online & offline devices. The default settings allocate 241 pages for configuration information, for *all* devices defined in the IOCDS. That includes offline devices which are defined for backup, network addresses, and miscellaneous 3270s etc. Issue the CP Q MONITOR command, and look at the MONITOR SAMPLE configuration size to see what is allocated. If you run CP3KVMXT, and it says the D1R7 or D1R9 records are missing, the problem is because there were too many device configuration records for the allocated Monitor Sample Configuration space. You will need to either enable monitoring for class dasd only, or enable i/o monitoring for specific devices, or redefine the monitor sample configuration space with more room.

[2] on January 1st, 2018, z/VM 6.4 will be the only release supported.

Starting with z/VM 6.2, the default monitor configuration space is much larger (4096 pages, instead of 241), so the size of the monitor sample configuration space would rarely be a problem.

Using Monwrite To Save Monitor Data in a File

If you want to save monitor information in a file, you will need a userid to connect to the monitor and use the MONWRITE utility to write out the data. This does not need to be the same userid that will, later, run the CP3KVMXT utility. It does not need anything higher than class G authority, but it will need special statements in its directory entry to be able to load the MONDCSS segment and connect to the monitor service. It needs a virtual storage size that will not conflict with MONDCSS, and it needs the following 2 statements to connect to the monitor:

```
IUCV *MONITOR MSGLIMIT 255
NAMESAVE MONDCSS
```

Typically, the monitor is started and stopped from a scheduler program such as WAKEUP, running on an operations utility userid.

Once the monitor is enabled, start up the MONWRITE machine, so that it can write monitor data out from the monitor DCSS into a file. Typically, the monwrite machine is started and stopped by the same WAKEUP machine used to start and stop the monitor.

From the userid which will run CP3KVMXT, link to the Monwrite disk. Make sure the Monview and CP3KVMXT program files and are present (they can be on any accessed disk).Start the CP3KVMXT utility. Refer to Command Syntax later in this doc for syntax. In the parmfile or as the first argument to CP3KVMXT, you should specify the input monitor data fileid. This too can be automated by a WAKEUP type utility userid, to start up CP3KVMXT once the monitor has been stopped.

Progress messages are issued at interval and duration ends, and after every 5000 records processed. The utility will end with a message about where the new EDF file was written.

Using the same process as when running CP3KVMXT with realtime monitor data, you can create an EDF from the intermediate results if the final EDF was not created for some reason.

Send the EDF file to the IBMer or Business Partner performing the Capacity Analysis.

For more information on setting up a Monwrite userid, please refer to “Chapter 9. Monitoring Performance Using CP Monitor” in SC24-5999 z/VM Performance Guide (see Appendix B for the URL of the z/VM Publications library).

Estimating the Space Needed for a Monitor Data File

Important: If you are running a z/VM systems with multiple guests, and are running z/VM 6.1 or earlier, the default MONCDSS size and monitor parameters are probably too small.

The sheer volume of z/VM Monitor data generated on a standard system is surprising, and for a system with complex guests and farms of dasd, it becomes important to take steps to collect just the desired monitor data. Tuning the running environment of CP3KVMXT is also important when you are running with realtime monitor data because all monitor records are processed, whether they are useful or not. Please review the section entitled "Speeding up CP3KVMXT" in Appendix B if you will be using realtime monitor data.

The amount of space needed for the monitor data depends on many things, including the number of users, dasd devices, partitions, the complexity of the guests, and the length of the CP monitor interval. Please refer to Table 1 on page 13 for storage requirement examples from 4 actual cases. The best way to estimate how much space the monitor data will be needed for your system is to collect data for 15 minutes and extrapolate from there.

	2817-718	2817-705	2097-717	2097-747
z/VM version	6.1	5.4	6.1	6.2
Total RDEV	3032	6992	2929	4118
Max Logged Users	46	69	43	4
Real engines	29	15	29	56
LCPs this image	6	5	6	5
partitions	9	17	7	4
Guests w/ multiple engines	7	43	5	0
# cache dev	1126	975	1048	10
CP Monitor Interval (min)	1	1	1	5
collection period (hrs)	1,0	15.7	1.0	0.5
dasd space (4k blks)	**31254**	**1054520**	**31993**	**13316**
3390 cyl per hour	**173.6**	**373.1**	**177.7**	**148.0**

Table 1. Monitor Data storage requirement examples

CP3KVMXT Installation and Setup

Setting up a userid to run CP3KVMXT

There are 2 basic ways to run CP3KVMXT: using Monitor Data which has been written to a file, or reading data directly from the Monitor DCSS in main storage. The requirements for the userid are different depending on how you want to run the utility.

Userid requirements for processing saved monitor data

Because monitor data processing can be a lengthy process, most sites set up a separate userid for CP3KVMXT, although it is possible to use an existing userid. The same userid that collected the data for the saved monitor data file could be used to process it into an EDF file, but generally it is better to have a different userid, so that the MONWRITE utility can be collecting current monitor data, while CP3KVMXT is concurrently processing the previous day’s data.

This userid will require read access to the mdisk or SFS directory containing the saved monitor data. It will also require a reasonably large virtual storage size. A 64meg virtual machine should be enough, but 128 meg may be needed for a system with many DASD devices. The virtual storage size may have to be adjusted up or down so as not to conflict with the reserved addresses of saved systems such as CMS.

CP3KVMXT processing uses more virtual storage than it does disk storage, but for a quick estimate, divide the number of DASD rdevices in half to get the approximate number of 4k blocks of writable disk space required. For example, a system with 612 DASD addresses would need 306 * 4k = 1.2meg of writeable disk space for a single run of the extract utility. This estimate is for the space required to process a monitor data file, and does *not* include the space for the monitor data itself. Please refer to the "Estimating the Space Needed" section for disk space requirements for monitor data.

Userid requirements for processing realtime monitor data

A dedicated userid is required to process realtime monitor data (data which is being generated real-time and which has *not* been saved to a file). This userid needs to be authorized to connect to the monitor. Authorization is granted by statements in the userid's directory entry that look something like this:

```
 IUCV *MONITOR MSGLIMIT 255
NAMESAVE MONDCSS
```

It will also require a reasonably large virtual storage size. A 64meg virtual machine should be enough, but 128 meg may be needed for a system with many DASD devices. The virtual storage size may have to be adjusted up or down so as not to conflict with the address of the MONDCSS. It also may need to be assigned some relative or absolute share to ensure that it can keep up with the monitor data as it is being generated. If it is not able to keep up, monitor data pages will be re-written before CP3KVMXT has a chance to read them. CP3KVMXT will stop when this happens.

Software Prerequisites

1. CMS Pipelines – this is part of the z/VM operating system, and should already be available to all VM users. If you can run this command, you already have access:
   pipe q version
2. FTP or a file transfer program of 3270-emulator. You will need this to install the software prerequisites and to download the finished EDF file.
3. VMARC - VMARC is a data compaction and file archiving utility for VM/CMS. The attributes of files compacted with VMARC are protected during file transfer, which is critical to machine readable formats such as in z/VM Monitor records. VMARC is available from the VM Downloads website.
4. MONVIEW provides tools for reading z/VM monitor data. Monview is available from the VM Downloads website. If you do not have the Rexx Compiler installed, see the installation step below about MDATTRAN.
5. CP3KVMXT to read monitor files and produce EDF files. See details below.

CP3KVMXT Installation

Please see the url above for step-by-step instructions, or refer to installation instructions on the http://www.vm.ibm.com/download website and in the CP3KVMXT PACKAGE file for the most current information. But basically, here's what you do:

- Download VMARC: http://www.vm.ibm.com/download/vmarc.module

- Download Monview: http://www.vm.ibm.com/download/packages/monview.vmarc
- Download CP3KVMXT: ftp://ftp.software.ibm.com/software/mktsupport/techdocs/CP3KVMXT.vmarc
- upload vmarc.module, monview.vmarc, and CP3KVMXT.vmarc in binary mode to the VM system where CP3KVMXT will be running. Make sure to keep the filetype, don't let the upload program append "BIN" to it.
- On the VM system, run the following commands:

```
PIPE < VMARC MODULE A | deblock cms | > VMARC MODULE A
VMARC UNPK MONVIEW VMARC A = = A (replace
VMARC UNPK CP3KVMXT VMARC A = = A (replace
```

- On the VM system, REXXCOMP to see if the Rexx compiler is installed. If it says "Unknown CP/CMS command", then the userid running CP3KVMXT does *not* have access to the Rexx Compiler. If so, then you need to swap the compiled and interpreted versions of Monview (substitute the correct fm, if not A):
  - RENAME MDATTRAN REXX A MDATTRAN CREXX A
  - RENAME MDATTRAN SREXX A MDATTRAN REXX A

CP3KVMXT Options

The PARMS file is a way to run CP3KVMXT with preset parameters, without having to respond to input prompts. It also is the *only* way to specify any option other than duration length, start date/time, and # of durations to process.

Autorunning with the PARMS Parameter file

Once you have done a test run, and have properly estimated the dasd space, virtual storage, and other runtime requirements, then you can use this option to help automate data collection. Edit the sample CP3KVMXT PARMS file, and save it with a meaningful filename, like the same filename as the output EDF. Later, after testing, you can use the command **CP3KVMXT (PARM *filename*** to start the cp3kvmxt run from the profile exec or some other automation tool. A leading * in the parm file marks a comment line. Lines containing parameters cannot also contain comments.

- **enterprise** identifies the case study, and appears on the cover letter of zCP3000 output reports. The name of the customer, or the data center name, is the sort of thing you should use here.
- **input** - is a required parameter. It identifies the source of the input monitor data. This can be a filename, filetype, and filemode (enclosed in double quotes), or the keyword "MONDCSS", to indicate the real-time monitor data is to be used. **If the FILELIST option is set to "YES", then then input file specified here is actually a list of monitor data files to be processed in this run.**
- **output** – is normally commented out. If specified, it identifies the output file. The default output file has the same filename as the input, filetype = 'EDF', and the filemode of the first accessed r/w minidisk.

- **duration** sets the interval that will be reported in zCP3000 reports. It is normally longer than, and must be a multiple of, the VM Monitor sample interval. **Note:** if you intend to load model data from more than one partition, only edf's with identical durations and overlapping sample date will be loaded.
- **FILELIST –** by default is "NO", but when it is set to "YES", then the file specified by the input parameter will contain a list of monitor data fileids.
- **processio** - normally is "YES", but when it is set to "NO", CP3KVMXT will ignore i/o configuration and sample records in the collected monitor data. If you are not going to be looking at dasd i/o anyway, this will speed things up. Caveat: there is one system health check rule that looks at the device types for paging devices; this will be N/A (not appear) in the System HealthCheck if i/o data has not been collected.
- **Syncstart** – is by default, set to "Yes". This means CP3KVMXT should delay the first measured interval until a "normal" time. So if you start the monitor at 11:58 with syncstart = "Yes", the first interval will start at 12:00, not 11:58. zCP3000 will round timestamps to the nearest 5 minutes, but if monitor started collecting at 11:57, it will round to 11:55, and will not be able to match up with an EDF starting at 12:00. This is important if you will be loading data from more than one partition into zCP3000. **Note: this option is deprecated, and will eventually disappear. Please use the starttime option instead.**
- **durlimit** - is normally commented out. This limits the number of durations (zCP3000 intervals) that will be processed.
- **startdate** - is normally commented out. If specified, CP3KVMXT will skip monitor data until if finds sample data on or after this start date. Please note that the date format is USA style : mm/dd/yyyy.
- **starttime** - is normally commented out. If specified, CP3KVMXT will skip monitor data until if finds a sample on or after this start date.

Running CP3KVMXT

Because of the amount of z/VM Monitor data generated on a standard system, it is important to take steps to collect just the desired monitor data and to ensure that the process runs correctly. There are 2 basic ways to run CP3KVMXT: using Monitor Data which has been written to a file, or reading data directly (like from a firehose) from the Monitor DCSS in main storage as it's being written by the CP Monitor service. For most installations, it will be simpler and easier to have monwrite write then data to disk, then have cp3kvmxt process it later. You can specify the input either from a command line prompt or from the PARMS file, but whichever method you use for the production run, you should start with a small test run.

Start With a Small Test Run

A useful strategy is to plan to collect a 15 minute sample, preview it, use it to estimate disk space requirements, and send it all of the way through the process, including downloading it to the PC, and loading into zCP3000. It's much easier to find and fix process problems when there's not so much data involved, when you're not under pressure to process that day's data so that MONWRITE has room to write out the next day's data.

Preview the Saved Monitor File

The preview function shows key information about a saved monitor file. This can help you make decisions about what duration to use, how much space to allocate, and whether this really is a valid monitor data file. Output from the preview function looks like this:

```
System VMOSP5      - processing 151485 Monitor records
All configuration records required for zCP3000 are present.
CPUMF Hardware monitor records have been found in this data.
Data collection started: 20150325 13:27:55.306645750
         last timestamp: 20150325 13:56:34.197470375
   CP monitor interval = 1 minute.
   SCP version/release = Z/VM 6.3.0 Service level 1401
            Hipervisor = LPAR
             CPU model = 2964-736
     Devices monitored = 1676
       Logged on users = 34
  Users w/ active CMMR = 2
   General purpose CPs = 36
           IFL engines = 48
          zIIP engines = 12
          zAAP engines = 0
           ICF engines = 0
Sampling enabled for: System Monitor Memory User Processor IO Virt Net ApplData
Sampling disabled for:
Events enabled for: Monitor Memory User Processor IO Virt Net ApplData
Events disabled for: Scheduler Seeks
```

Invoke the preview like this:

Preview *monFileName monFileType fm*

The results of the preview are displayed at the console and saved in a file called *monFileName PREVIEW fm*

Decide on your Runtime Parameters

There are a number of factors that will influence the run, the most important one being whether a capacity planning study will be done of this z/VM system alone, or whether the modeler will be considering, say, the 5 other production partitions on the same machine. Each system can be studied individually, but to consider the effect that partitions many have on one another when sharing engines, coupling facility links, and processor cache (RNI Relative Nest Intensity), it is necessary to look at a concurrent period of time.

- If the capacity planning study will include EDFs from more than one system, then they all have to use the same duration, cover about the same period of time, and have duration timestamps that, when rounded to the nearest 5 minutes, have the same values. So for example, if SYSA and SYSB both have 30 minute durations, but SYSA started collecting at 10am, and SYSB started collecting at 10:10am, then SYSA's durations will be 10:00, 10:30, 11:00, while SYSB will have duration timestamps of 10:10, 10:40, 11:10. The model data from SYSA and SYSB cannot be loaded together in zCP3000, and the systems will have to be looked at individually. An option called “SyncStart”, which skips monitor records until a more regular start time is reached, is enabled by default.
- When choosing the duration, consider how much long a period of time you are collecting data for, so that you will end up with a reasonable number of samples in the capacity planning model. If you collect 1 hour of data, and use a 30 minute interval, you will end up with (at most) 2 data points on a line graph; not very interesting. If you collect 24x7, and use a 15 minute duration, you will have 672 data points on the line graph, which will be close to unreadable.
- The selected duration must be a multiple of the CP Monitor sampling interval. If the monitor interval is 2 minutes, then you cannot have a 15 minute duration.
- Some zCP3000 options can only be selected by running the utility from the CP3KVMXT PARMS file. The only options that can be specified via prompts on the command line are: duration length, start time, and # of durations to process. Disabling I/O processing, or disabling the syncStart option can only be done in the PARMS file.

Cookbook: Using Realtime Monitor Data:

The following procedure is for running CP3KVMXT realtime. The procedure for using saved monitor data follows this section.

Note: running against realtime monitor data can be the more complicated of the two methods, especially if the site does not already run the monitor. If tuning is needed, you will need to work closely with the site system programmer to size the dcss and try the various suggestions for optimizing CP3KVMXT's performance.

1. Plan to run a test against 15 minutes of monitor data and reviewing the results and the dasd space requirements before starting the real collection.
2. Review the setup requirements in the "Setting up CP3KVMXT" section and ensure that the CP3KVMXT userid has sufficient disk space, virtual storage, the prerequisite software packages installed, and has been granted authorization to connect to the MONDCSS segment.
3. Make sure that the userid which will run CP3KVMXT has been authorized to connect to the Monitor DCSS. Authorization is granted by these statements in the userid's directory entry:

   ```
   IUCV *MONITOR MSGLIMIT 255
   NAMESAVE MONDCSS
   ```

4. If there is a virtual storage addressing conflict with the Monitor DCSS, you will have to adjust the virtual storage size of the userid running CP3KVMXT. You will know this right away when you start running CP3KVMXT if you get the message *DMSDCS343E Storage in range 01400000-02FFFFFF for MONDCSS in use*. If this happens, try increasing the size of your virtual machine (using CP Q V STOR, then DEF STOR ___M). The max virtual storage setting in the directory entry may have to be increased to relieve the conflict.
5. If you will be using a parm file to supply runtime prompts, edit the CP3KVMXT PARMS file, change the runtime parameters, and save it with a meaningful filename (filetype must be PARMS). The meaning of the various parms is explained in comments in the CP3KVMXT PARMS file. By default, CP3KVMXT will prompt for parameters. Parms supplied on the command line for the input and output fileids, will override the corresponding parm in the PARMS file.
6. If monitor data is not normally gathered at this site, then you must enable and start the CP monitor from an E class userid. Please refer to the Monitor Requirements for detail. The monitor should run during core business hours, including both peak and non-peak times. From the class E userid, you can CP Q MONITOR to see its status. You should see something like this:

   ```
   MONITOR EVENT ACTIVE    BLOCK    4     PARTITION    3584
   MONITOR DCSS NAME - MONDCSS
   CONFIGURATION SIZE       68 LIMIT        1 MINUTES\
   CONFIGURATION AREA IS FREE
   USERS CONNECTED TO *MONITOR - MONWRITE
   ```

```
MONITOR   DOMAIN ENABLED
PROCESSOR DOMAIN ENABLED
STORAGE   DOMAIN ENABLED
SCHEDULER DOMAIN DISABLED
SEEKS     DOMAIN DISABLED
USER      DOMAIN ENABLED
   ALL USERS ENABLED
 I/O       DOMAIN ENABLED
THE FOLLOWING DEVICES ARE ENABLED:
2000-203F  2300-247F  2540-257F  2600-263F  2C00-2EFF  4000-427F
4300-44FF  4800-4848  4880-48C8  4900-4948  4980-49C8  4A00-4BFF
5000-50FF  5300-53FF  6300-637F  6400-647F  7000-77FF  8420-842F
8440-844F  AC00-AFFF  B300-B3FF  B700-B7FF  B900-BAFF  BD00-BDFF
BF00-C1FF  D800-D853  D900-D953  E900-E93F

APPLDATA  DOMAIN DISABLED
MONITOR SAMPLE ACTIVE
              INTERVAL    5 MINUTES
              RATE     1.00 SECONDS
MONITOR DCSS NAME - MONDCSS
CONFIGURATION SIZE      241 LIMIT        1 MINUTES
CONFIGURATION AREA IS FREE
USERS CONNECTED TO *MONITOR - MONWRITE
MONITOR   DOMAIN ENABLED
SYSTEM    DOMAIN ENABLED
PROCESSOR DOMAIN ENABLED
STORAGE   DOMAIN ENABLED
USER      DOMAIN ENABLED
ALL USERS ENABLED
I/O       DOMAIN ENABLED
THE FOLLOWING DEVICES ARE ENABLED:
2000-203F  2300-247F  2540-257F  2600-263F  2C00-2EFF  4000-427F
4300-44FF  4800-4848  4880-48C8  4900-4948  4980-49C8  4A00-4BFF
5000-50FF  5300-53FF  6300-637F  6400-647F  7000-77FF  8420-842F
8440-844F  AC00-AFFF  B300-B3FF  B700-B7FF  B900-BAFF  BD00-BDFF
BF00-C1FF  D800-D853  D900-D953  E900-E93F
APPLDATA  DOMAIN DISABLED
Ready; T=0.01/0.04 20:26:08
```

7. Start the CP3KVMXT utility to create a 15 minute test EDF file, referring to Command Syntax later in this doc for syntax, and specifying MONDCSS as the input source. Do not use the CP3KVMXT PARMS file for the test run.
8. To stop data collection, any class E userid can issue the following command to manually stop the monitor with this command so that it can gracefully finish writing out interval data:
   ```
   CP MONITOR STOP
   ```
   Note: The monitor needs to be stopped only for an instant to complete the

Monitor Data file. You can start the monitor back up again right away for other applications that use it (i.e., VMRTM or FCONX).

9. You may also have a scheduler/WAKEUP machine issue the CP MONITOR STOP after a certain interval.
10. After addressing any problems, start the CP3KVMXT utility to create the full EDF file, specifying MONDCSS as the input source.

    - If you specified that CP3KVMXT use the parameter file by using the PARMFILE option, it will pick up runtime parameters from there and override them with any command line options. Otherwise, you will see several prompts, the first of which is to enter a title for the study, including the customer name and perhaps a date.
    - If you did not specify the CP3KVMXT parm file, the utility will ask you for a DURATION. This is the period of time that a SAMPS record in zCP3000 will represent. You should pick a value which is a multiple of the CP Monitor interval and will generate a reasonable number of data points on a utilization graph (for example, 4 hours of data and a duration of 15 minutes would generate a graph with 16 data points). For an 8 hour measurement period, 30 minutes would be a good value for the DURATION.

11. Progress messages are issued at interval and duration ends, and after every 5000 records processed. The utility will end with a message about where the new EDF file was written. If you want to end CP3KVMXT earlier than the time you specified (via prompt or parmfile), you can do this by hitting the <enter> key once, letting it sit in VM READ for 2 minutes, and then hitting the <enter key> again. This will time out the MONDCSS connection. Entering HX will also work, but will likely abend CMS and require that the userid re-ipl.
12. If the final EDF was not created for some reason (i.e., you HX'd it), you can create an EDF from the intermediate results file with the following command[3]

    ```
    CP3KVMXT fn SAMPS A ( restart
    ```
13. You can see a console of the run in a file with the same filename and filemode as the output EDF, but with the filetype CONSOLE.
14. Download the EDF file in ascii mode and send it to the IBMer or Business Partner performing the Capacity Analysis. Please make sure the record length formatting has been preserved, and that the lines do not “wrap”.

[3] Refer to the CP3KVMXT Syntax section for more detail.

Cookbook: Using Saved monitor data:

There are 2 basic ways to run CP3KVMXT: using Monitor Data which has been written to a file, or reading data directly from the Monitor DCSS in main storage. The following procedure is for using saved monitor data. The procedure for running CP3KVMXT with realtime data precedes this section.

Note: running against saved monitor data is usually the simpler of the two methods, as long as sufficient disk space for the monitor file is available.

1. Plan to run a test against 15 minutes of monitor data and reviewing the results and the dasd space requirements before starting the real collection.
2. Review the setup requirements in the "Setting up CP3KVMXT" section and ensure that the CP3KVMXT userid has sufficient disk space, virtual storage, the prerequisite software packages installed, and has access to a monitor data file.
3. If you will be using a parm file to supply runtime prompts, edit the CP3KVMXT PARMS file, change the runtime parameters, and save it with a meaningful filename (filetype must be PARMS). The meaning of the various parms is explained in comments in the CP3KVMXT PARMS file. By default, CP3KVMXT will prompt for parameters. Parms supplied on the command line for the input and output fileids, will override the corresponding parm in the PARMS file.
4. If monitor data is not normally gathered at this site, then you must enable and start the CP monitor from an E class userid. Please refer to the Monitor Requirements for detail. The monitor should run during core business hours, including both peak and non-peak times. From the class E userid, you can CP Q MONITOR to see its status. You should see something like this:

```
MONITOR EVENT ACTIVE    BLOCK     4     PARTITION     3584
MONITOR DCSS NAME - MONDCSS
CONFIGURATION SIZE       68 LIMIT         1 MINUTES\
CONFIGURATION AREA IS FREE
USERS CONNECTED TO *MONITOR - MONWRITE
MONITOR   DOMAIN ENABLED
PROCESSOR DOMAIN ENABLED
STORAGE   DOMAIN ENABLED
SCHEDULER DOMAIN DISABLED
SEEKS     DOMAIN DISABLED
USER      DOMAIN ENABLED
   ALL USERS ENABLED
 I/O       DOMAIN ENABLED
THE FOLLOWING DEVICES ARE ENABLED:
2000-203F  2300-247F  2540-257F  2600-263F  2C00-2EFF  4000-427F
4300-44FF  4800-4848  4880-48C8  4900-4948  4980-49C8  4A00-4BFF
5000-50FF  5300-53FF  6300-637F  6400-647F  7000-77FF  8420-842F
```

```
8440-844F  AC00-AFFF  B300-B3FF  B700-B7FF  B900-BAFF  BD00-BDFF
BF00-C1FF  D800-D853  D900-D953  E900-E93F

APPLDATA  DOMAIN DISABLED
MONITOR SAMPLE ACTIVE
             INTERVAL    5 MINUTES
             RATE     1.00 SECONDS
MONITOR DCSS NAME - MONDCSS
CONFIGURATION SIZE      241 LIMIT        1 MINUTES
CONFIGURATION AREA IS FREE
USERS CONNECTED TO *MONITOR - MONWRITE
MONITOR   DOMAIN ENABLED
SYSTEM    DOMAIN ENABLED
PROCESSOR DOMAIN ENABLED
STORAGE   DOMAIN ENABLED
USER      DOMAIN ENABLED
ALL USERS ENABLED
I/O       DOMAIN ENABLED
THE FOLLOWING DEVICES ARE ENABLED:
2000-203F  2300-247F  2540-257F  2600-263F  2C00-2EFF  4000-427F
4300-44FF  4800-4848  4880-48C8  4900-4948  4980-49C8  4A00-4BFF
5000-50FF  5300-53FF  6300-637F  6400-647F  7000-77FF  8420-842F
8440-844F  AC00-AFFF  B300-B3FF  B700-B7FF  B900-BAFF  BD00-BDFF
BF00-C1FF  D800-D853  D900-D953  E900-E93F
APPLDATA  DOMAIN DISABLED
Ready; T=0.01/0.04 20:26:08
```

5. Start the CP3KVMXT utility to create a 15 minute test EDF file, referring to Command Syntax later in this doc for syntax, and specifying a 5 minute duration and a maximum 3 duration limit to be processed. Do not use the CP3KVMXT PARMS file for the test run.
6. After addressing any problems, start the CP3KVMXT utility to create the full EDF file, specifying MONDCSS as the input source.

   - If you specified that CP3KVMXT use the parameter file by using the PARM *filename* option, it will pick up runtime parameters from there and override them with any command line options. Otherwise, you will see several prompts, the first of which is to enter a title for the study, including the customer name and perhaps a date.
   - If you did not specify a PARM file, the utility will ask you for a DURATION. This is the period of time that a SAMPS record in zCP3000 will represent. You should pick a value which is a multiple of the CP Monitor interval and will generate a reasonable number of data points on a utilization graph (for example, 4 hours of data and a duration of 15 minutes would generate a graph with 16 data points). For an 8 hour measurement period, 30 minutes would be a good value for the DURATION.

8. Progress messages are issued at interval and duration ends, and after every 5000 records processed. The utility will end with a message about where the new EDF file was written. If you want to end CP3KVMXT earlier than the time you specified (via prompt or parmfile), you can do this by hitting the <enter> key once, letting it sit in VM READ for 2 minutes, and then hitting the <enter key> again. This will time out the MONDCSS connection. Entering HX will also work, but will likely abend CMS and require that the userid re-ipl.
9. If the final EDF was not created for some reason (i.e., you HX'd it), you can create an EDF from the intermediate results file with the following command[4]

   ```
   CP3KVMXT fn SAMPS A ( restart
   ```
10. You can see a console of the run in a file with the same filename and filemode as the output EDF, but with the filetype CONSOLE.
11. Download the EDF file in ascii mode and send it to the IBMer or Business Partner performing the Capacity Analysis. Please make sure the record length formatting has been preserved, and that the lines do not "wrap".

Recovering from problems

If CP3KVMXT ran for a while, but you saw the following message:

```
RC=12 from line 64    'readto xrecdata' /* branch to error when rc=12 EOF (normal) */
EDF creation was unsuccessful. Checkpoint data set not created.
```

The most common reason for this is that you asked for a duration that was longer than the data actually collected (data collection needs to be for an extra interval or two in addition to the amount you intend to collect). For example, you would see this if you collected 10 minutes of data and asked for a 15 minute duration.

If an EDF was not created, but it seemed like CP3KVMXT ran for a while, look in the SAMPS file which has the same filename as your input. If this has a new timestamp, and there are several interval records in the SAMPS file, then you can create an EDF with this many intervals, using the checkpoint records, running the following command against that SAMPS file:

```
CP3KVMXT fn SAMPS fm ( RESTART
```

[4] Refer to the CP3KVMXT Syntax section for more detail.

What to do with the output

EDF files can be used by either zPCR or zCP3000.

zPCR

zPCR typically uses one EDF to pick up the LPAR configuration information for the entire machine. It can also be used to identify the CPUMF workload for *this* partition. You don't typically need to collect as long a period of time for zPCR as you would for zCP3000. **If this EDF is for a zPCR study, then one EDF (at a reasonably representative time), is probably good enough**. The file won't be that big, so you typically can attach it to an email to send to whoever is going to be using it with the zPCR tool.

zCP3000

For zCP3000, however, you will typically collect a week's worth of data, so you need to think about naming conventions for the output. You might also want to consider using the Filelist option to process multiple monitor data files in one cp3kvmxt run (please see the section "What if you collect hourly monitor data files?"). Zip all of the EDF files together before sending them to your IBM or Business Partner representative.

Where to upload data

zPCR is available to customers as well as IBM and Business Partners, but zCP3000 studies can be done only by IBM employees and IBM Business Partners. Because of the size of the files, and because of FTP security restrictions, the process for uploading data will vary, so please work with the person who will be doing the study for you on transferring the EDF files needed for the study.

Appendix A: Syntax

Scroll down to Syntax examples for a quick reference.

Only one argument is required: the input data source. By default, the user will be prompted to specify

1. the interval length (duration) that will appear in capacity planning reports
2. the start date and time, defaulting to the beginning
3. the number of durations to process

The SYNCSTART and PROCESSIO options default to “yes” and can only be specified if running CP3KVMXT using the PARM file option.

```
                                . outputFileid .
>>--CP3KVMXT-+- monfileid -----+--------------+----------+---|
             '- MONDCSS -------'              '- ( option ) -'
             '- inputFileList -'
             '- fn SAMPS fm   -'

Options:
     .-RESTART-.
|----+---------+-----------|
     '-PARM filename ------'
     '-FILELIST -----------'
```

Monitor input source:

monFileid : *filename filetype filemode* of a saved monitor data file. Monitor data files *always* have a record length of 4096. This may be on a filemode accessed in a read-only mode.

MONDCSS : says to connect directly to the monitor service to read realtime performance data from the MONDCSS saved segment.

inputFileList : *filename filetype filemode* of a file that contains a list of monitor data files (fn ft fm). You must also specify the FILELIST option.

SAMPS : *filename SAMPS filemode* is part of the checkpoint data from an unsuccessful CP3KVMXT run, from which you want to create an EDF. There will also be a CECDATA with the same filename, and BCUDATA, if i/o data was collected. The RESTART option must also be specified to create the EDF from a checkpoint.

outputFileid : *filename filetype filemode* of the file where you want the output to be written. The filemode, if specified, must be a filemode accessed in a read/write mode.

> **Default output filename :** the filename of the monitor data file
> **Default output filetype : EDF**
> **Default output filemode :** the first filemode accessed in r/w mode

RESTART: says that the CP3KVMXT run did not end successfully, but that you would nevertheless like to create an EDF file from the checkpoint data collected during the run. The SAMPS fileid must be specified as the data input for this option.

PARM *filename*: says to use the *filename* PARMS file to supply runtime parameters. This will look on all filemodes and use the first one it finds.

FILELIST: says that the specified input source is actually a list of monitor data files to be processed.

Syntax Examples

CP3KVMXT *fn ft fm*

> will run CP3KVMXT against the monitor data file indicated by *fn ft fm*, and will write the output into *fn* EDF A. You will be prompted to supply the zCP3000 reporting interval (the duration), and if there are any limits on when reporting should start and stop.

CP3KVMXT (PARM *filename*

> will read the *filename* PARMS file and will run CP3KVMXT against the monitor data file identified in the parm file, and write the output to the file identified by the parm file. All options, including the delayed start times, duration length, and a maximum number of durations to process, are specified in the CP3KVMXT PARMS file. If you are automating the data collection process, you should probably use this method.

CP3KVMXT *fn ft fm* (FILELIST

> Says that the fileid contains a list of monitor data files that should be processed into a single output EDF, which will have the name ***fn EDF fm.***

CP3KVMXT fn SAMPS fm (restart

> Will build an EDF from an intermediate SAMPS checkpoint results file.

Appendix B: Problems and Solutions

Problems before you even run CP3KVMXT

CP3KVMXT and its pre-reqs are downloaded to a PC then uploaded to a z/VM system. It is necessary that this be done in binary (image) mode so that no ASCII->EBCDIC translation of the executable code is done. CMS Pipelines, which is part of z/VM, is also required. Please review the installation procedure on page 14 of this document.

Symptom: VMARC UNPK says Invalid header for compacted file.
Explanation: possibly something went wrong in the file transfer. Upload it again, and make sure it is being transferred in binary mode. A VMARC compacted file that has been properly uploaded should be fixed record length 80, the first 4 characters will be ":CFF", and you will see the name of the first compacted file following that in clear text.

Symptom: CP3KPREQ : There is a problem with the prerequisites

CP3KVMXT requires the VMARC (zip-like utility for z/VM) and MONVIEW (monitor data utilities) packages. You must install VMARC first, and then use it to unpack MONVIEW and CP3KVMXT.

Problems running CP3KVMXT that happen right away

Symptom: DMSSTT002E File * not found

Explanation: Either one of the parts of the the CP3KVMXT package are missing, or the input monitor file was not found. Please download CP3KVMXT again, then re-install the package: VMARC UNPK CP3KVMXT VMARC A = = A (REPLACE

Symptom: FPLREX562E Alternate exec processor EAGRTPRC; return code -3

Explanation: the rexx compiler is not installed. If your site has the rexx compiler, then access it and add an access statement to the userid's profile. Otherwise, you will need to rename two files shipped with the MONVIEW package. Monview ships both compiled (CREXX) and uncompiled (SREXX) versions of MDATTRAN. Simply rename the MDATTRAN.REXX to CREXX, and the MDATTRAN.SREXX to REXX.

Symptom: DMSDCS343E Storage in range 01400000-02FFFFFF for MONDCSS in use.

Explanation: You must redefine the size of your virtual storage so that it does not conflict with the MONDCSS segment. I.e., in the above example, the user's virtual storage size was set to 32M, and MONDCSS tried to load into the area from 20-50M. DEF STOR 64M then re-ipling CMS fixed the problem.

Symptom: FPLSMG319E Not authorized to communicate with *MONITOR

Symptom: DMSDCS283E The MONDCSS saved segment could not be loaded;

return code 449 from SEGMENT LOAD.

Explanation: The directory entry for the userid running cp3kvmxt needs 2 statements like this:
IUCV *MONITOR MSGLIMIT 255
NAMESAVE MONDCSS[5]

Abends: Rexx, syntax, and command errors

Symptom: Arithmetic overflow/underflow

Symptom: Incorrect call to routine

Explanation: there is a bug in the cp3kvmxt pipeline or rexx code. Frequently, it’s a problem that’s already been solved. Check the download site (see Websites: in Appendix E) and make sure you are running the current version of cp3kvmxt. It is typically updated every other month.

Problems with CP3KVMXT before the end of the first duration

Symptom: ERROR: there were not enough monitor records to create an EDF.
Explanation: End-of-file occurred before there were enough records to make up a duration. Also, CP3KVMXT may have skipped some intervals in order to get to a more normal starting time (ie, 08:00 instead of 07:58). Use the Preview function to make sure there is enough data in the monitor data file for the duration you chose. If there were a lot of skipped records, you can specify the NOSYNC option and run from the parm file.

Symptom: ERROR: CP Monitor sample interval is too high.
Symptom: ERROR: Duration is shorter than the CP Monitor sample interval.
Symptom: ERROR: The duration should be set to a multiple of the interval.
Explanation: The CP monitor sample interval should be less than the CP3KVMXT duration, and the duration should be a multiple of the interval. Interval data is accumulated into durations, and the durations should be the length that you want to see in the capacity planning graphs.

Symptom: * sample domain disabled

Explanation: CP3KVMXT requires monitor sampling be enabled for processor, user, and storage domains. I/O sampling and any type of event monitoring are optional. Please review

[5]The NAMESAVE statement is needed only if the MONDCSS was defined as a restricted segment.

CP Monitor Requirements on page 4 for more information on CP monitor settings.

Symptom: * event domain disabled

Explanation: CP3KVMXT requires user event monitoring to be enabled, but most other event monitoring is not required. It is useful, however, to have event monitoring enabled so that CP3KVMXT can figure out what is happening when devices or processors are varied on/offline. Please review

CP Monitor Requirements on page 4 for more information on CP monitor settings.

Symptom: CP3KVMXT completes, but **Memory Configuration record is missing.**
Explanation: If you see this record missing but you did not see a message about the System Configuration Data being missing, then most likely your MONDCSS is too small for the number of devices that you enabled for the monitor service.

Symptom: CP3KVMXT completes, but there's a message about "missing records".
Explanation: CP3KVMXT ends with the message "The following records were missing or not processed-", and lists one or more types of records. This may be a result of monitor records being overwritten in memory before they could be processed, or it could be because the monitor sample configuration area was too small for the number of devices being monitored. Please review the section in the user's guide entitled "CP3KVMXT Input Data Requirements**:**".

Problems with CP3KVMXT that happen after running a bit

Symptom: runtime message "Error: sysid has changed"

Explanation: More than one D1R4 Monitor Sample Configuration Record was read in the monitor data, and the D1R4 records refer to different sysids. This probably happened if you concatenated monitor data from different z/VM images into one monitor data file, or used the FILELIST option to process multiple monitor data files but one of them was from a different sysid. CP3KVMXT will only process one sysid at a time.

Symptom: HCPMOV6274I sample data messages … have been purged.
Symptom: FPLSMG313E IPRCODE Message was purged .. on IUCV instruction.
Explanation: While running with realtime monitor data, CP3KVMXT ended because of these messages. The messages mean that CP3KVMXT could not keep up with the monitor. There must be less data or more speed or both. Refer to "Speeding up CP3KVMXT" on page 34 for suggestions of things you can do to speed things up.

Symptom: "FPL122E" or "Insufficient storage"
Symptom: "Machine storage exhausted";
Explanation: there is not enough virtual storage defined for this userid to run the extract. Use the Q V STOR and DEF STOR xxM command to define a larger virtual

storage size. If your userid is already at the max, you will need to have the system administrator authorize you for a larger virtual storage size.

Symptom: CP3KVMXT completes with RC=28 from line 124 'EXEC SAMP2EDF'.

Explanation: If there doesn't seem to be any other error, then the most common reason for this is when

1. there is not enough monitor data to make a full duration (ie, you collected monitor data for 20 minutes but asked for a 30 minute duration).
2. all of the monitor data collected occurs before the specified start date.
3. the monitor interval sample (ie, 5 minutes) combined with when the monitor sample starts (ie, 12:03), combined with the synchronize option, means that it never reaches the desired starting point.

Preview the data to make sure the duration and start date are suitable, and if the monitor sampling interval is greater than 1 minute, use the parm file and change the **syncStart** parameter to "NO".

Symptom: CP3KVMXT completes, but no EDF file is created.
Explanation:. If a large file with a filetype of DEBUG exists on the A disk, then CP3KVMXT probably did not complete successfully. If that is the case, then please send it to Gretchen Frye at **frye@us.ibm.com**, along with an explanation of the problem. Checkpoint data is kept as the utility runs, so it is possible to generate an EDF from the part of the monitor data that was successfully processed. Use the restart option to build an EDF from intermediate results (see above).

Symptom: CP3KVMXT completes, but no EDF file is created, and it seems to take a long time doing something after the last duration has been processed.
Explanation: It may have failed in the step that creates the EDF out of the checkpoint data. If the SAMPS file is very large, it may run out of virtual storage when processing the checkpoint data. If the SAMPS and BCUDATA files were created (check the timestamp), and you can see that the checkpoint is large (ie, larger than the DEBUG file), try the restart option:

```
CP3KVMXT fn SAMPS fm ( RESTART
```

If that doesn't work, try increasing the virtual storage of the machine and use the restart option again. If that doesn't work, then it may be necessary to re-run CP3KVMXT and use the start and end parameters to create multiple, smaller EDFs.

Symptom: CP3KVMXT completes, an EDF is created, but there is no CPUMF data.
Explanation: The PREVIEW file will tell you if there are any D5R13 records in the monitor data. If PREVIEW says there is no CPUMF data, then please see the requirements for collecting CPUMF data on page 9.

Symptom: REXX error 35 in line nnn: Invalid expression
Symptom: REXX error 40 in line nnn: Incorrect call to routine
Symptom: REXX error 41 in line nnn: Bad arithmetic conversion

Explanation: Something unexpected was seen in the data or there is an error in the CP3KVMXT program. Try the following steps:

1. Make sure the monitor file you are running against is actually a CP Monitor file. Monitor data files are in hex, in a FB4096 file (it will look unreadable).
2. Run a Preview against the monitor data file. If this does not complete successfully, there is something wrong with that monitor data.
3. If looks like a program error, please verify at http://www.vm.ibm.com/download/packages/descript.cgi?CP3KVMXT that you are running the current version of the code. The code version number appears in the console messages. If not current, then download the current and try again.
4. If the current version of the code is abending, then I will want to see the larger of the two DEBUG files that is created and the console file that was created. Please attach the debug file and screenshot/console to an email and send to the author, Gretchen Frye, at frye@us.ibm.com. If I need to see the source monitor data, I will send instructions for transmitting it.

Informational Messages

These messages may be seen at the end of a CP3KVMXT run. They are informational messages which may influence but would not prevent the creation of an EDF file.

Symptom: Info: n SAN devices attached to n guests systems were found.
Explanation: These are storage devices directly attached to a user running a guest operating system, and driven by the guest operating system itself. z/VM does not collect performance measurements from these devices, so none of the activity to these devices will be included in any report that shows device or control unit based activity. This could be a significant part of the guest i/o activity, but there is no way to know this. We just want you to know it's there, so you can decide whether the remaining i/o activity is worth looking at.

Appendix C: Speeding up CP3KVMXT:

This section applies both to realtime MONDCSS input and saved monitor data files, but it is much more critical for realtime input because the connection to the monitor ends if CP3KVMXT can't keep up with the monitor. When processing realtime monitor input, CP3KVMXT, after reading each set of records, replies to the monitor service that it has read them, so that the monitor service can reuse that space when all of the connected users have responded. If it does not reply by the time the monitor service wants to reuse those pages, the monitor takes the pages and issues an HCPMOV6274I error message. The CMS Pipeline connection to the monitor ends the connection when this happens, issuing the FPLSMG313E error message.

1. **Monitor less stuff**. You can cut back on what it being monitored and reduce the number of records being produced, which will reduce the number of records that CP3KVMXT must read through.
   - Unless a performance monitor is using it for some specific purpose, the scheduler and seeks domains should *not* be enabled. CP3KVMXT does not use these records. *CP Q MONITOR* and you should see the following both for event handling and sampling:

     ```
     SCHEDULER DOMAIN DISABLED
     SEEKS     DOMAIN DISABLED
     ```

   - Enable the monitor only for DASD class I/O:

     ```
     CP MONITOR SAMPLE ENABLE I/O CLASS DASD
     ```

   - Even better, enable only a specified device address range. Online but inactive dasd still gets sampled every minute, and even for offline dasd, configuration records are built and read for all devices in the IOCDS. These are all of the "Reading monitor record 10000 ... D1 system configuration" records that you see at the beginning of a CP3KVMXT run. If you can identify a few dozen or fewer rdevice addresses of dasd that are active, not just online, then you can enable monitoring on just those devices like this:

     ```
     CP MONITOR SAMPLE ENABLE I/O DEVICE 3008-300A 3007 3103-
     3105 3174 317B 31A8 3200-321F
     ```

   - Or enable only the CP-owned volumes, where system paging and spooling go. This will tell you nothing about application-driven I/O, but it's easier than figuring out which devices to monitor.

     ```
     CP MONITOR SAMPLE ENABLE I/O VOLume CPVOL
     ```

2. **ProcessIO = “NO” to ignore i/o records.** If there are just too many dasd devices, or you are not really interested in the read hit ratio or device service time, you can use the PROCESSIO = “NO” parameter in the *filename* PARMS file to have CP3KVMXT ignore the i/o records. This will speed up processing considerably, like in half or more, because CP3KVMXT will no longer be keeping and updating the 90 variables tracked for each actuator and control unit every minute or so of the sampling interval.
3. **Ensure that the MONDCSS segment is the right size for your system**. The default size of 3MB may not be big enough, especially for guest-hosting systems with lots of DASD. The smaller it is, the more quickly it has to round-robin and reuse the space, and the more quickly CP3KVMXT has to process it. Follow the guidelines for sizing MONDCSS in the z/VM Performance Guide on http://www.vm.ibm.com/library/
4. **If your site has the Rexx compiler** (REXXCOMP MODULE is the compiler), try renaming and compiling CP3KVMXT REXX. The compiled CP3KVMXT must have a filetype of REXX in order to execute. It won’t make any difference to compile the CP3KVMXT EXEC, because it is mainly a front end and does not execute much code.
5. **If you see any HCPMOV6274I messages**, look carefully at those messages and note whether the problem occurred with “event” or “sample” data. Increase the size and limit of the config on the CP Monitor command for “event” or “sample” so that the records are retained a little longer.
6. **Consider increasing the CP Monitor interval size**. This will reduce the sampling frequency, and thereby reduce the amount of data collected.

Appendix D: EDFI Field Descriptions

The EDF file is a fixed block file with a record length of 80, with very restrictive formatting requirements. The z/OS equivalent field is provided as a matter of interest. Some fields are exactly the same in z/OS and z/VM (the LPAR related fields, for instance, are provided by the underlying PR/SM microcode). Others are similar in intent but not necessarily measuring the same thing (for instance, most workload related fields).

Table 1: HEAD macro – header information

Head	Header Section	Type (length)	Domain / Rec	VM Source Fieldname	MVS equivalent
ENT	Enterprise Name	char(50)	-	User	User
Source	Cp3kvmxt version and date	char(23)	-	generated	
VER	Version only	char(4)	-		
SMFDSN	Input data files	char	-		

Table 2: CEC macro – information about this mainframe

CEC	Description	Type (length)	Domain / Rec	VM Source Fieldname	MVS equivalent
CECID	CPC Identifier	Char	-	“CEC” & serial number	User
CMIND	Index of current Vector CPC in CPUMODV	Vector			
CPUMOD	CPU Model	Char	D1R4	MTRSYS_SYSMTYPE, user prompt	SMF70MOD SMF70VER
CPUMODV	CPU Model Names	Multiple text strings	D1R4	MTRSYS_SYSMTYPE	SMF70MOD SMF70VER
CPV	General Purpose Processors	Vector	D0R17	SYTCUM_LCUPTYPE	
HWCMODV	CPC hardware model (books)	String	D1R4	MTRSYS_STSI111	SMF70MOD SMF70HWM
HWIND	Index of current model in HWCMODV	Number			
IFLV	IFL engines	Vector	D0R17	SYTCUM_LCUPTYPE	
ICFV	Coupling Facility processors	Vector	D0R17	SYTCUM_LCUPTYPE	
PRV	Total # engines	Vector	D0R17	SYTCUM_LCUPCPCT	SMF70BNP
SR	CPU serial #	Number	D1R5	MTRPRP_PFXIDSER	SMF70SER
SPX	MCL for VM65396 is installed	Binary	D0R19	SYTSYG_PFXSHLAV	
SUPVR	Supervisor	Char	-	generated	
VC	Machine /model version code	Char	D1R5	MTRPRP_PFXIDVER	SMF70VER
ZAAPV	IFA or AAP processors	Vector	D0R17	SYTCUM_LCUPTYPE	
ZIIPV	IIP processors	Vector	D0R17	SYTCUM_LCUPTYPE	

Table 3: SYS information about this z/VM image and LPAR information

SYS	System Image	Type (length)	Domain / Rec	VM Source Fieldname	MVS equivalent
ASRES	Shared Address Space CS resident	Vector (frames)	D3R14	STOASI_ASCCTPRS + STOASI_ASCCTPRG	n/a
BIT	64 bit mode indicator	0 or 1	D1R4	MTRSYS_CALESAME	SMF70EME
CAI	Capacity Adjustment Indicator	Vector	D1R18	MTRCCC_SYSCAI	SMF70CAI
CCR	Capacity Adjustment Reason	Vector	D1R18	MTRCCC_SYSCCR	SMF70CCR
CPI (type:CP)	CPUMF: Cycles per Instruction	Vector	D5R13	PRCMFC_COUNTERSB0/ B1	SMF113
CRYNU	Number of Coprocessors	Vector	D5R10	PRCAPM_VAR_DATA count	SMF7023N
CRYRA	Total crypto rate	Vector	D5R10	PRCAPM_CMB1_C0	R7023CO
CRYTM	Total crypto	Vector	D5R10	PRCAPM_CMB1_T0	R7023TO*, R7023SF
CRYTY	Coprocessor Type	Vector	D5R10	PRCAPM_CT	R7023CT
CRYUT	Crypto Utilization	Vector		Derived form CRYTM, CRYNU	n/a
CS	LPAR defined CS	MB	D1R7	MTRMEM_RSAGSTOR	SMF71TFC +SMF71FIN
CSAVAILV	available frames below 2GB	Vector (MB)	D0R3	SYTRSG_RSAAVAIL	SMF71TFC +SMF71FIN
DASDIOV	Dasd SSCH rate	Vector	D1R6, D6R3	MTRDEV_RDEVCLAS, IODDEV_SCMSSCH	SMF74SSC
DCSS	Shared Program Area	Vector (frames)	D3R3	STOSHR_ASCCTPRS+ STOSHR_ASCCTPRG	n/a
DPA	Pageable Area	Vector (MB)	D0R3	SYTRSG_RSAPGABL+ SYTRSG_RSALGFRM	n/a
ES	LPAR defined ES	MB	D1R17	MTRXSG_SYSXTSIZ	SMF71OLE
ESAVAILV	available frames above 2GB	Vector (MB)	D0R3	SYTRSG_RSA2GAV2	SMF71TFC +SMF71FIN
FCPI (type:CP)	CPUMF: Estimated CPI from Finite cache/memory	Vector	D5R13	PRCMFC_COUNTERS	SMF113
GHZ (type:CP)	CPUMF: CPU Speed (cycles/usec)	Vector	D5R13	PRCMFC_CCFCPUSP	SMF113
GMTOF	offset GMT	+/- hh:mm	D1R4	MTRSYS_SYSZONE	
IFANF	Normalization Factor for zAAP	Vector	D1R4	MTRSYS_CPUCAPAB, MTRSYS_SCPCAPAB	R723NFFI
IIPNF	Normalization Factor for zIIP	Vector	D1R4	MTRSYS_CPUCAPAB, MTRSYS_SCPCAPAB	R723NFFS
L15P (z10, type:CP)	CPUMF: %source from Level 1.5 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113

L15P (z10, type:IFL)	CPUMF: %source from Level 1.5 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113
L1MP (type:CP)	CPUMF: Level 1 Miss	Vector	D5R13	PRCMFC_COUNTERS (B2+B4)/B1	SMF113
L2LP (z10, type:CP)	CPUMF: %source from on-book L2 cache	Vector	D5R13	PRCMFC_COUNTERS (z10)	SMF113
L2P (type:CP)	CPUMF: %source from Level 2 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113
L2RP (z10, type:CP)	CPUMF: %source from off-book L2 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113
L3P (type:CP)	CPUMF: %source from Level 3 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113
L4LP (type:CP)	CPUMF: %source from on-book L4 cache	Vector	D5R13	PRCMFC_COUNTERS (z196)	SMF113
L4RP (type:CP)	CPUMF: %source from off-book L4 cache	Vector	D5R13	PRCMFC_COUNTERS (z196)	SMF113
LCPI (type:IFL)	CPUMF: Cycles per Instruction	Vector	D5R13	PRCMFC_COUNTERSB0/B1	SMF113
LFCPI (type:IFL)	CPUMF: Estimated CPI from Finite cache/memory	Vector	D5R13	PRCMFC_COUNTERS	SMF113
LGHZ (type:IFL)	CPUMF: CPU Speed (cycles/usec)	Vector	D5R13	PRCMFC_CCFCPUSP	SMF113
LL1MP (type:IFL)	CPUMF: Level 1 Miss	Vector	D5R13	PRCMFC_COUNTERS (B2+B4)/B1	SMF113
LL2LP (z10, type:IFL)	CPUMF: %source from on-book L2 cache	Vector	D5R13	PRCMFC_COUNTERS (z10)	SMF113
LL2P (type:IFL)	CPUMF: %source from Level 2 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113
LL2RP (z10 only, type:IFL)	CPUMF: %source from off-book L2 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113
LL3P (type:IFL)	CPUMF: %source from Level 3 cache	Vector	D5R13	PRCMFC_COUNTERS	SMF113
LL4LP (type:IFL)	CPUMF: %source from	Vector	D5R13	PRCMFC_COUNTERS (z196)	SMF113

	on-book L4 cache				
LL4RP (type:IFL)	CPUMF: %source from off-book L4 cache	Vector	D5R13	PRCMFC_COUNTERS (z196)	SMF113
LMEMP (type:IFL)	CPUMF: % sourced from Memory	Vector	D5R13	PRCMFC_COUNTERS (z196)	SMF113
LPAR	Partition names	Vector	D0R16	SYTCUP_LCUPNAME	SMF70LPM
LPARNO	index of this sysid in the LPAR array	Number	D0R16	SYTCUP_CALPTIS	SMF70PTN
LPCAPn	is partition capped? (1 or 0)	Vector	D0R16	SYTCUP_LCUCCAPP	SMF70VPF, SMF70CAP
LPMODE	Processor Configuration Mode	“ZVM” or “LINUX”	D1R4	MTRSYS_SYSCMODE	n/a
LPP*TMn[6]	CPU time (seconds)	Vector	D0R16	SYTCUP_LCUCACTM	SMF70PDT
LPPR*n[6]	Engines online	Vector	D0R16	SYTCUP_LCUPCPCT	SMF70ONT
LPRBS (type:IFL)	CPUMF: % Problem State	Vector	D5R13	PRCMFC_COUNTERS	SMF113
LPRWTAn	GCP absolute capping	Vector	D0R16		
LPRWTA*[6]	Absolute capping	Vector	D0R16		
LPW*n[6]	partition weight; L=IFL, P=zIIP, I=ICF, Z=zAAP	Vector	D0R16	SYTCUP_LCUCWGHT	SMF70MIS, SMF70BPS, SMF70MAS
LPWTn	partition weight for CP	Vector	D0R16	SYTCUP_LCUCWGHT	SMF70MIS, SMF70BPS, SMF70MAS
LRNI (type:IFL)	CPUMF: Relative Nest Intensity	Vector	D5R13	PRCMFC_COUNTERS	SMF113
LSCPL1M (type:IFL)	CPUMF: Sourcing Cycles per Level 1 Miss	Vector	D5R13	PRCMFC_COUNTERS (processor dependent)	SMF113
MDCCS	Minidisk Cache in main storage	Vector (frames)	D0R3	SYTRSG_TCMMAIN	n/a
MDCES	Minidisk Cache in ES	Vector (frames)	D0R14	SYTXSG_HCPMDCNE	n/a
MDCRHR	MDC read hit ratio	Vector	D0R14	SYTXSG_HCPMDCIA / SYTXSG_HCPMDCIA	n/a
MEMP (type:CP)	CPUMF: % sourced from Memory	Vector	D5R13	PRCMFC_COUNTERS (z196)	SMF113
NF	Normalization Factor to show	Vector	D1R4	MTRSYS_CPUCAPAB, MTRSYS_SCPCAPAB	n/a

[6] Where * means C=CP, L=IFL, P=zIIP, I=ICF, Z=zAAP, and n is the partition number

	IFL equivalent in CP				
NONPG	# non-pageable frames	Vector (frames)	D0R3	SYTRSG_RSANONPG	n/a
NSAMPS	Number of samples	Number	-	generated	generated
PAGEDEVS	Paging Device Types	Vector	D1R8, D1R6	MTRDEV_RDEVDVID	n/a
PAGEDS	Page Datasets (count)	Vector	D1R8	MTRPAG_CALTYPE = ‘PAGE’	n/a
PAGESLOT	Paging Area on Aux (4k pages)	Vector	D1R8	MTRPAG_RDCPCYL * MTRPAG_CALCYLNO	n/a
PAGEUTIL	Page Slot Utilization	Vector	D3R4, D1R8	STOASP_CALPAGE / (MTRPAG_RDCPCYL * MTRPAG_CALCYLNO)	n/a
PAGEV	Page Operations	Vector	D0R1	SYTSYP_PLSIOPR + PLSIOPW	n/a
PGES	Unallocated ES used for paging	Vector (MB)	D1R17, D0R14, D4R3,	Calculated: ES minus MDC minus sum(user XSTOR)	n/a
PGFROMES	CS page from ES	Vector	D0R5	SYTXSP_PLSPGOUT	SMF71RES
PGTBL	User VStor Page Tables	Vector (frames)	D3R14	STOASI_ASCCTPRS + STOASI_ASCCTPRG	n/a
PGTOES	CS page to ES	Vector	D0R5	SYTXSP_PFXPGIN + PLSPGIN	
PRBS (type:CP)	CPUMF: % Problem State	Vector	D5R13	PRCMFC_COUNTERS	SMF113
RMFINTL	CP monitor sampling interval	Number (min)	D1R9	MTRSPR_INTERVAL	
RNI (type:CP)	CPUMF: Relative Nest Intensity	Vector	D5R13	PRCMFC_COUNTERS	SMF113
SCP	SCP	Char	D1R4	MTRSYS_HCPCPEPP	SMF70MVS
SCPCS	Total Resident Nucleus	MB	D1R7	MTRMEM_HCPMM4	SMF71FIN+ASR+ALP+AVP
SCPL1M (type:CP)	CPUMF: Sourcing Cycles per Level 1 Miss	Vector	D5R13	PRCMFC_COUNTERS (processor dependent)	SMF113
SMTEFF*[6]	Multithreading capacity	Vector	D0R2	SYTPRP_CAL_CAPBYTYPE	SMF70CF
SMTEFFM*[6]	Multithreading max capacity	Vector	D0R2	SYTPRP_CAL_MAXCAPBYTYPE	SMF70CF
SMTEFFTD*[6]	Multithreading thread density	Vector	D0R2	SYTPRP_CAL_AVGTDBYTYPE	SMF70CF
SPXDA	VM65396 is installed	Vector	D1R4 or D1R31	calculated	
SSINM	SSI Cluster Name	String	D1R25	MTRSSI_SYSPLXNM	n/a
SSINAMES	SSI Members	String	D1R25	MTRSSI_PMSSYSNM	n/a
SVCLVL	Service level	String	D1R4	MTRSYS_HCPCPEID	n/a
SYSID	System ID	Char	D1R4	MTRSYS_SYSTMID	SMF70SID

TLBC*	CPUMF: TLB1 cycles per TLB miss; C=CP, L=IFL	Vector	D5R13	PRCMFC_COUNTERS (processor dependent)	SMF113
TLBM*	CPUMF: TLB1 miss %; C=CP, L=IFL	Vector	D5R13	PRCMFC_COUNTERS (processor dependent)	SMF113
TLBP*	CPUMF: PTE % of all TLB1 misses; C=CP, L=IFL	Vector	D5R13	PRCMFC_COUNTERS (processor dependent)	SMF113
TV	System T/V ratio (total/virtual)	Vector	D0R2	(PFXUTIME+PFXTMSYS) / PFXPRBTM	n/a
VERSION	SCP level	Char	D1R4	MTRSYS_HCPCPEID	SMF70RLS
WC	Wait Complete	0 or 1	D0R16	SYTCUP_LCUCWCPL	SMF70VPF, SMF70WSA

Table 5: WORK – user/workload group samples

WORK	Workload Samples	Type (length)	Domain / Rec	VM Source Fieldname	MVS equivalent
CMMR	Collaborative Memory Mgmt	Vector	D4R3	USEACT_VMDMAACT	n/a
LOCKED	Pages locked in CS	(Vector (frames)	D4R3	USEACT_VMDCTPVL	n/a
SVM	Service Virtual Machine	Binary	D1R15	MTRUSR_VMDSVMST	n/a
VDIO	Sum I/Os to all virtual disk	Vector	D3R17	STOVDK_QDIIOCNT	n/a
VDISK	Size of VDisks summed	Vector (512byte blocks)	D3R17	STOVDK_CALSIZE	n/a
VDRES	Resident VDisk pages	Vector (frames)	D3R14	STOASI_ASCCTPRS+ STOASI_ASCCTPRG	n/a
VSTORV	Virtual Storage	Vector (MB)	D4R3	USEACT_ASCDEFSZ	n/a
WCPUTM	GCP type CPU seconds	vector	D4R3	USEACT_VMDTTIME	SMF72CTS+ SMF72STS+ R723Cnnn
WCSV	resident pages	Vector (frames)	D4R3	USEACT_VMDCTPVR + USEACT_VMDCTPVG	SMF72FT1,2
WDESC	Description	Char	D4R3	USEACT_VMDUSER	user parm
WESV	pages in ES	Vector (frames)	D4R3	USEACT_VMDCTXBK + USEACT_CALXSTOR	
WEXCPV	Wkld Dasd IO Requests	Vector	D4R3	USEACT_VMDVDSCT	SMF72ITS +R723CIOC
WIFLV	IFL type CPU seconds	Vector	D4R3	USEACT_VMDTTIME	n/a
WIOV	SSCH count	vector	calc	IODDEV_SCMSSCH	SMF72IRC +R723CIRC
WPAGEV	Wkld Paging	Vector	D4R3	USEACT_CALCPPGR +CALCPPGW	SMF72PIN +R723CPIN
WSPXDB	Speculative Execution Protection	Vector	D1R15 D4R3	MTRUSR_VMDSHLON	
WSSV	Projected working set	Vector (frames)	D4R3	USEACT_VMDWSSPR	n/a
WVCN	Virtual GCP type engines	Vector	D4R3	USEACT_VMDPUTYP	n/a

WVLN	Virtual IFL type engines	Vector	D4R3	USEACT_VMDPUTYP	n/a
WVTM	Problem State Time (GCP)	Vector	D4R3	USEACT_VMDVTIME	n/a
WVTML	Problem State Time (IFL)	Vector	D4R3	USEACT_VMDVTIME	n/a
XSTORV	Attached ES	Vector (MB)	D4R3	USEACT_CALXSTOR	n/a

Table 6: BCU dasd i/o information

BCU	Basic Configurable Unit for DASD	Type (length)	Domain / Rec	VM Source Fieldname	MVS equivalent
BCUID	BCU Identification	char	D6R4	IODCAD_CALDATA	User Parm
CTYPE	CU type	char	D1R6	MTRDEV_RDEVCUID, MTRDEV_RDEVCUMN	User Parm/ SMF74CU
BCUDASD1	Dasd type (online devices only)	char	D1R6	MTRDEV_RDEVDVID	User Parm/ SMF74DEV
BCUDASDN	# units this type (online devices only)	num	D6R3	counted	computed
CACHE	Available cache	num (MB)	D6R4	device hardware	UserParm/ CSCONF
NVS	Non-volatile storage	num (MB)	D6R4	device hardware	UserParm/ CSCONF
NOAD	# addresses under this BCU	num	D6R3	counted	From BCU MAP
BCUIO	Total I/O Rate	num[7]	D6R3	IODDEV_SCMSSCH	SMF74SSC
BCURESP	Average Response Time	num		computed	computed
BCUCONN	Average Connect Time	num	D6R3	IODDEV_SCMCNTIM	SMF74CNN
BCUDISC	Average Disconnect Time	num	D6R3	IODDEV_SCMDDTIM	SMF74DIS
BCUPEND	Average Pend Time	num	D6R3	IODDEV_SCMFPTIM	SMF74PEN
BCUQUE	Average IOS Queue Time	num	D6R3	IODDEV_HFCTIO	SMF74QUE

[7]Time fields in the BCU are in seconds.

BCUSKEW	Maximum device busy to average .	num		computed	computed
BCUR	Read/Write Ratio8	num	D6R4	computed	CRR:"Total (Cache) R/W Ratio"
BCUH	Ratio: Read Hits9 / All Reads	0< num <1	D6R4	IODCAD_CALDATA	CRR:"Total Read H/R"
BCUW	Ratio: Fast Write10 Hits / All Writes	0< num <1	D6R4	IODCAD_CALDATA	CRR:"Total F/W H/R"
BCUG	Ratio: Sequential DASD to Cache / All I/O	0< num <1	D6R4	IODCAD_CALDATA	CRR: "DASD to Cache Transfers - Sequential" divided by "Total I/O Requests"
BCUIOV	I/O Rate per sample.	vector	D6R3	IODDEV_SCMSSCH	SMF74SSC
BCURESPV	Avg Response Time per sample44	vector	D6R3	computed	computed
BCUCONNV	Avg Connect Time per sample	vector	D6R3	IODDEV_SCMCNTIM	SMF74CNN
BCUDISCV	Avg Disconnect Time per sample	vector	D6R3	IODDEV_SCMDDTIM	SMF74DIS
BCUPENDV	Avg Pend Time per sample	vector	D6R3	IODDEV_SCMFPTIM	SMF74PEN
BCUQUEV	Avg IOS Queue Time per sample	vector	D6R3	IODDEV_HFCTIO	SMF74QUE

[8]BCU time fields include all online devices under that BCU, whether they are cached or not.
[9]Cache Reads include Search/Read requests for Normal, Sequential, and Fast Write I/Os.
[10]Cache Writes include Write requests for Normal, Sequential, and Fast Write I/Os

BCURV	Read Write Ratio per sample	vector	D6R4	computed	CRR:"Total (Cache) R/W Ratio"
BCUHV	Read Hit Ratio per sample	vector	D6R4	IODCAD_CALDATA	CRR:"Total Read H/R"
BCUWV	Fast Write Hit Ratio per sample	vector	D6R4	IODCAD_CALDATA	CRR:"Total F/W H/R"
BCUGV	Sequential Stage Ratio per sample.	vector	D6R4	IODCAD_CALDATA	CRR: "DASD to Cache Transfers - Sequential" divided by "Total I/O Requests"

Table 7: BCU Path information

PATH	BCU Path Data	Type (length)	Domain / Rec	VM Source Fieldname	MVS equivalent
PID	Path ID	Char chpid	D6R3	IODDEV_RDEVLPM	User
PTYPE	Channel Type	P parallel E ESCON F FICON		class B CP command Q CHPID chpid TYPE	SMF73ACR
PBUSYV	Path Busy	vector	D0R20	CMG1: CMCPBT_CPC /elapsed	SMF73BSY
PBBY	Ficon Bus Busy	Vector	D0R20	CMG2: CMCBC_CPC/ CSCCMCMB	SMF73TBC and SMF73MBC

Table 8: Actuator information

ACT	Actuator Data	Type (length)	Domain / Rec	VM Source Fieldname	MVS equivalent
SID	Subchannel ID	num (hex)	D6R3	IODDEV_RDEVLPM	User
V	Volser[11]	Char	D6R3	IODDEV_RDEVSER	SMF74SER
A	Address	num (hex)	D1R6	MTRDEV_RDEVDEV	SMF74NUM
T	DASD Type	char	D1R6	MTRDEV_RDEVCUID, MTRDEV_RDEVCUMN	SMF74DEV
R	I/O Rate	num	D6R3	IODDEV_SCMSSCH	SMF74SSC
SDS	standard deviation for service	num	D6R3	computed	computed
Q	IOSQ	num	D6R3	IODDEV_HFCTIO	computed
P	Pend Time	num	D6R3	IODDEV_RDEVFPTIM	SMF74PEN
D	Disconnect Time	num	D6R3	IODDEV_RDEVDDTIM	SMF74DIS
C	Connect Time	num	D6R3	IODDEV_RDEVCNTIM	SMF74CNN
DS	Minidisks Defined	num	D6R3	IODDEV_RDEVLCNT	SMF74NDA
SDR	standard deviation for response	num	D6R3	computed	computed
RWR	Read Write Ratio	num	D6R3	computed	CRR:"Total (Cache)46 R/W Ratio"
RDHT	Read Hits / All Reads	num	D6R4	IODCAD_CALDATA	CRR:"Total Read H/R"
FWHT	Fast Write Hits / All Writes	num	D6R4	IODCAD_CALDATA	CRR:"Total F/W H/R"
SQSTG	Sequential DASD to Cache / All I/O	num	D6R4	IODCAD_CALDATA	CRR: DASD to Cache Transfers - Sequential divided by Total I/O Requests
PC	Percent Cacheable	num	D6R3	IODCAD_CALDATA	CRR: Total Cacheable I/Os divided by

[11]Volser is not unique, since a device attached to a guest may have any volser (or null).

					Total I/O Requests
ST	Actuator Status	N - Caching Activiated, DASD FW Allowed, D - Caching Deactivatd DASD FW Deactivatd C - Caching Activated, DASD FW Deactivatd F - Caching Deactivatd DASD FW Allowed	D6R4	IODCAD_CALDATA	CRR: "Device Status".

Appendix E: Resources

Websites:

IBM VM Download Packages
http://www.vm.ibm.com/download/packages/

How to download a VMARC type package from VM Downloads
http://www.vm.ibm.com/download/#downvmarc

IBM WSC - Capacity Planning Support Tools menu
IBM Internal: http://w3.ibm.com/support/americas/wsc/cpsproducts.html
IBM BP: http://partners.boulder.ibm.com/src/atsmastr.nsf/WebIndex/PRS1762

z/VM Internet Library
http://www.vm.ibm.com/library/

Recorded Presentation: "2015 z/VM Data Collection for zPCR and zCP3000"

http://lt.be.ibm.com/weblectures/online/ltu49933/cp3kvmxt.mp4

Publications:

The **CP3KVMXT reference** (that you are reading right now) can be found at:
ftp://ftp.software.ibm.com/software/mktsupport/techdocs/cp3kvmxt.pdf

Illustrated Guide to Using VMARC

http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS3332

z/VM Data Collection Guide

http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS2875

zCP3000 Discussion Group:

http://groups.google.com/group/zcp3000

Contacts:

For non-technical questions and general support: Valerie Spencer

Internet email address: cpstools@us.ibm.com

Normal US business hours (EST): 301-240-2645

For level2/level3 technical issues: Gretchen Frye

Internet email address: frye@us.ibm.com

To send data to the **Testcase Data Exchange Service**, please review the Service User License Agreement at http://www-05.ibm.com/de/support/ecurep/service.html

then contact Gretchen for the upload procedure.

Return to the top