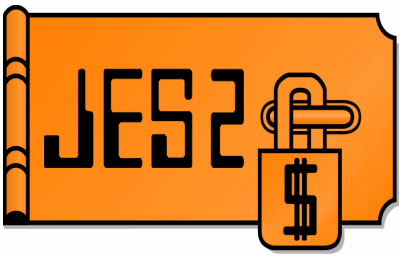
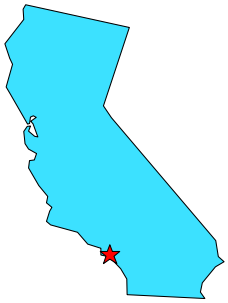


The JES2 Health Monitor



SHARE 102, Session 2657

Wednesday, Feb. 25, 2004

Chip Wood/Tom Wasik
JES2 Design/Development/Service
Poughkeepsie, NY

chipwood@us.ibm.com

Objectives

What's the problem?

- How to diagnose why JES2 is not responding to JES2 commands?
- A process is not functioning, what command will indicate why?

Solution

- New address space to examine JES2 activity
- Commands to summarize JES2 problems

Overview

- JES2 "Health" Monitor
  - JES2 base code beginning in **z/OS 1.4**
  - Not a "performance" monitor (yet)
  - Some basic performance information is tracked
  - Intent is to surface **VERY SEVERE** performance problems (JES2 not responding, etc.)
- Separate address space
  - Name is ***jesx*MON** (***jesx*** - subsystem name)
  - Started as part of JES2 initialization
  - Terminated on clean JES2 termination ($PJES2)
  - Automatically restarted monitor if failure
- New load module HASJ2MON

The JES2 Monitor is part of the JES2 z/OS 1.4 base, and is started automatically with JES2.
It's important to note that the intent of the JES2 Monitor is a "health" monitor, not a "performance" monitor. It is intended to address situations where JES2 is not responding to commands and the installation is unsure of what the problem is. It could be the result of a command that is just taking a long time to complete, or a bug in JES2, an EXIT, or some other code running in the JES2 address space. The monitor is intended to help identify the problem so the installation can take the correct action.
Though the monitor does collect data that could be useful in tuning JES2, that is not the intended purpose of the monitor. Monitoring performance parameters is a future consideration for the monitor.

Overview *(continued)*

- Multiple Subtasks each performing single function
  - Main task - starts and stops address space
  - Sampler - samples JES2 TCB and resources usage
  - Probe - examines samples and issues alerts
  - Command - processes operator commands
  - Very low overhead
- Message ids are 4 digit ($HASP9xxx)

Overview *(continued)*

- Sampler processing
  - Examines JES2 main task TCB/RB 20 times/sec
    - Looks for main task waits, loops, other delays
    - Records all unexpected MVS waits
  - Examines resource usage once per second
    - Tracks low, high, and average usage
    - Low, high, average reset top of every hour
    - Up to 72 hours of history are retained
    - Tracks same resources as HASP050 message
  - Sampler feeds probe logic

The sampler subtask gathers data about what the JES2 main task TCB, main PRB and current RB are doing. This is done by directly accessing storage in the JES2 address space. Based on the data obtained, the code determines if the main task is:

at its normal MVS wait in HASPNUC
at some other MVS wait
waiting for the local lock
non-dispatchable
waiting for a page fault resolution
running code normally
looping

The sampler also tracks data on resource usage. The resources monitored are currently the same as those reported on the HASP050 message. A list can be seen later in some of the command responses.

Overview *(continued)*

- Probe processing
  - Runs every 4 seconds
  - Based on sampler data reports JES2 main task

MVS waits	Paging waits
Local lock waits	Loops
Non-dispatchable	Long PCE dispatch
Busy - No MVS waits	
Not running - Not waking up from main WAIT	

  - Monitors JOB and BERT lock PCE waits
  - Monitors long hold of JES2 checkpoint

The probe subtask is looking at the sampler data and determining if there is a problem. It runs less frequently because it is looking for relatively long-term trends in the sampling data.
The checkpoint lock held condition is issued based on hold times. If the hold time is set high (intentionally or by mistake), this condition is not monitored.

Overview *(continued)*

- Probe actions
  - Action on event depends on duration
    - 0-n seconds, no action
    - n-x seconds, track event, display only with command
    - x+ seconds, issue alert message
    - Every 30 or 120 seconds, reissue alert message
  - Times (n and x) vary based on specific event
  - All clear message for main task/CKPT lock

The probe looks for conditions that may indicate a potential problem. It then examines how long the condition has existed. Based on the duration it will either:

ignore the condition if it has only been happening for a very short time

start tracking it as a potential problem. This creates a record in the monitor address space which can be displayed via the $JDJES command.

start alerting the condition. This causes a message to be issued informing the operator of the problem. The message may be repeated on a timer (with updated status information).

Events are grouped based on type. Two major groupings are main task events and checkpoint lock held events. When these transition from having had an alert to no tracks, an "all clear" message is issued.

Probe Examples

Probe example 1 (using test code)

```
  23.10.15  $tmonitor,qsuse,loop1=1
  23.10.29 *$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR
HASTDIAG+01F2FE
  DURATION-000:00:14.60 PCE-COMM     EXIT-NONE JOB ID-NONE
  COMMAND-$TMONITOR,QSUSE,LOOP1=1
  23.10.29 *$HASP9207 JES2 CHECKPOINT LOCK HELD
  DURATION-000:00:14.66
  23.11.01 *$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR
HASTDIAG+01F2FE
  DURATION-000:00:46.61 PCE-COMM     EXIT-NONE JOB ID-NONE
  COMMAND-$TMONITOR,QSUSE,LOOP1=1
  23.11.01 *$HASP9207 JES2 CHECKPOINT LOCK HELD
  DURATION-000:00:46.66
  23.11.33 *$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR
HASTDIAG+01F2FE
  DURATION-000:01:18.62 PCE-COMM     EXIT-NONE JOB ID-NONE
  COMMAND-$TMONITOR,QSUSE,LOOP1=1
  23.11.33 *$HASP9207 JES2 CHECKPOINT LOCK HELD
  DURATION-000:01:18.67
  23.11.41  $HASP468 MONITOR  OK
  23.11.41  $HASP9301 JES2 MAIN TASK ALERTS CLEARED
  23.11.41  $HASP9302 JES2 CHECKPOINT LOCK RELEASED
```

In this example, a long loop is generated while holding the JES2 checkpoint (the **$T MONITOR** command is not a real JES2 command but an internal test command to artificially cause problems detected by the monitor).

After 15 seconds, two highlighted alert messages are issued:

- $HASP9202 indicates that a loop was detected near a specific offset in HASTDIAG. This means that for a long period of time ALL of the samples indicate that the JES2 main task was running near this location. Also, since this is the command PCE, the specific command that is running at the moment is displayed.
- $HASP9207 indicates that the JES2 checkpoint has been held by this member for a long time.

About 30 seconds later, these two messages are DOM'ed and reissued with updated information. This occurs until the condition has cleared.

In this case, the loop completes and JES2 operation returns to normal. The last messages are DOM'ed and $HASP9301 and $HASP9302 are issued (not hilighted) to indicate that everything is back to normal

Probe Examples

Probe example 2 (using test code)

```
  12.03.10  $dscantest,abend
  12.03.10 *$HASP095 JES2 CATASTROPHIC ABEND.  CODE = S0C1
  12.03.10 *17 $HASP070 SPECIFY RECOVERY OPTIONS - ('RECOVER' OR
   'TERMINATE' OR 'SNAP' AND, OPTIONALLY, ',NODUMP')
  12.03.24 *$HASP9201 JES2 MAIN TASK WAIT DETECTED AT HASPTERM+0054D8
  DURATION-000:00:14.54 PCE-COMM     EXIT-NONE JOB ID-NONE
  COMMAND-$DSCANTEST,ABEND
  12.03.26  17,recover
  12.03.26 $HASP080 JES2 SYSTEM DUMP REQUESTED FROM HASPTERM 00178000 +
002E06
  12.03.37 $HASP9203 LONG PCE DISPATCH 186
  DURATION-000:00:27.61 PCE-COMM     EXIT-NONE JOB ID-NONE
  COMMAND-$DSCANTEST,ABEND
  12.03.41 $HASP9209 JES2 MAIN TASK NON-DISPATCHABLE AT HASPRAS +0010DA
  DURATION-000:00:31.20 PCE-COMM     EXIT-NONE JOB ID-NONE
  COMMAND-$DSCANTEST,ABEND
  $HASP073 RECOVERY SUCCESSFUL - NORMAL PROCESSING RESUMES
  $HASP9301 JES2 MAIN TASK ALERTS CLEARED
```

In this example, another command is issued which causes a recoverable ABEND.
After about 15 seconds, $HASP9201 is issued to indicate JES2 has been MVS waiting, in this case for the reply to the $HASP070.
After the WTOR is replied to, JES2 stops MVS waiting, but the process of taking a dump takes a long time. There's no current MVS wait and no discernable loop, so $HASP9203 is issued to indicate that a PCE has been dispatched for an extraordinarily long time.
SDUMP processing eventually makes the JES2 main task non-dispatchable, and that is reflected in the $HASP9209 message. Note that the DURATION in the message is the total time since the problems started, not since a particular flavor of the problem was exposed.
Again, in this case, JES2 recovers and resumes normal processing, at which time all the outstanding messages are DOM'ed and the "all clear" message is issued.

Probe Examples

Probe example 3 (using test code)

```
17.37.58  $tj1-*,loop1=1
17.38.14 *$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR
 HASTDIAG+01FC4E
DURATION-000:00:15.77 PCE-COMM     EXIT-NONE JOB ID-JOB00011
COMMAND-$TJ1-*,LOOP1=1
17.38.14 *$HASP9207 JES2 CHECKPOINT LOCK HELD
DURATION-000:00:15.87
17.38.46 *$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR
 HASTDIAG+01FC4E
DURATION-000:00:47.62 PCE-COMM     EXIT-NONE JOB ID-JOB00035
COMMAND-$TJ1-*,LOOP1=1
17.38.46 *$HASP9207 JES2 CHECKPOINT LOCK HELD
DURATION-000:00:47.68
.
.
.
17.40.43  $HASP9301 JES2 MAIN TASK ALERTS CLEARED
17.40.43  $HASP9302 JES2 CHECKPOINT LOCK RELEASED
```

In this example, a loop is introduced, similar to an earlier example. This time, however, it's a command that runs through the job queue. The JOB-ID field is updated each time the $HASP9202 message is issued to reflect the current job the processor is working on. In some cases (like this one), the JOB-ID field can be used to gauge the progress of a loop or long-running process.

Probe Examples *(continued)*

Probe Messages

```
$HASP9201 JES2 MAIN TASK WAIT DETECTED AT module+offset
$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR
module+offset
$HASP9203 LONG PCE DISPATCH
$HASP9204 JES2 MAIN TASK BUSY
$HASP9205 PCE WAITING FOR BERT LOCK
$HASP9206 PCE WAITING FOR JOB LOCK
$HASP9207 JES2 CHECKPOINT LOCK HELD
$HASP9208 JES2 MAIN TASK LOCAL LOCK WAIT AT module+offset
$HASP9209 JES2 MAIN TASK NON-DISPATCHABLE AT module+offset
$HASP9210 JES2 MAIN TASK PAGING WAIT AT module+offset
$HASP9211 JES2 MAIN TASK NOT RUNNING
$HASP9212 MVS NOT DISPATCHING JES2 MAIN TASK (OA06186)
```

Second and 3rd line as appropriate

```
DURATION-xx:xx:xx.xx PCE-xxxxxxxx EXIT-xxx JOB ID-xxxxxxxx
COMMAND-xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Command Processing

- All monitor commands start with the JES2 command prefix followed by a 'J'
- If the command is not a valid monitor command it is routed to JES2 (in case of an installation command that starts with 'J')
  - Command order not preserved
- No exits are called (no exit 5)
- Commands can only come from SVC 34 (command SSI) or JES2 automatic command
  - No NJE, RJE, init deck, readers

Monitor commands are intercepted in the SSI and directed to the monitor command processor. To do this, the SSI code needs an easy way to route commands to the monitor. The method chosen was to use a 1 character command prefix of 'J'. All commands that have the JES2 command prefix followed by a 'J' are sent to the monitor command subtask first. No exits are taken for these commands (except for the pre/post SAF exits 36 and 37). If the monitor does not recognize the command, it is routed to the JES2 address space for normal command processing. Because of this, commands starting with a 'J' may execute out of order from other commands issued by the same source.
Monitor commands are only allowed from the SVC 34 command SSI and JES2 automatic commands. Sources such as NJE, RJE, internal readers, and the init deck are not allowed to issue monitor commands. However, monitor commands can be the object of $VS commands so they can be issued wherever a $VS can be used.

Command Processing *(continued)*

- CONDEF CMDNUM=, RDIRAREA=, and DISPMAX= do not apply
- Only one command per SVC 34
  - Cannot separate with JES2 command separator (';')
- Spaces and comments (/* */) are ignored
- L= is supported to direct response to out of line area or specific console
- RACF profiles are of the form
  - *jes2***MON**.*action.object*
  - *jes2* is the monitored subsystem name

There are some JES2 settings that do not apply to monitor commands. Monitor commands are not limited by **CONDEF CMDNUM=.** There is no limit as to the number of monitor commands that can be queued. Also the output of the commands is not limited by **CONDEF DISPMAX=**. Though you can direct the output of the command using the L= operand, the default area **CONDEF RDIRAREA=** is not supported.
Since these commands are not supported by the normal JES2 command processor, there is no support for the JES2 command separator ';'. Only one monitor command per SVC 34 request.
As with other JES2 commands, spaces and comments (/* */) are ignored.
RACF calls are made for all commands. Format of the RACF entity name protecting commands is similar to what is used for other JES2 commands. The first qualifier is the same as the monitor address space name.

$J D MONITOR

$J D MONITOR

- *jesx*MON.DISPLAY.MONITOR (READ)
- Displays monitor task and module status information

```
$HASP9100 D MONITOR
NAME     STATUS       ALERTS
-------- ------------ ------------------------
MAINTASK ACTIVE
SAMPLER  ACTIVE
COMMANDS ACTIVE
PROBE    ACTIVE
$HASP9102 MONITOR MODULE INFORMATION
NAME     ADDRESS  LENGTH   ASSEMBLY DATE  LASTAPAR LASTPTF
-------- -------- -------- -------------- -------- --------
HASJMON  06A2A000 00001088 07/10/02 04.34 NONE     NONE
HASJSPLR 06A2C000 00002838 07/10/02 04.35 NONE     NONE
HASJCMDS 06A2F000 00003050 07/10/02 04.34 NONE     NONE
```

The $J D MONITOR command displays the status of the monitor itself. It includes two messages:

$HASP9100 displays the status of each of the monitor subtasks.

$HASP9102 displays module information for each monitor module, similar to what the $D MODULE command displays for other JES2 modules.

$J STOP

$J STOP

- *jesx*MON.STOP.MONITOR (CONTROL)
- Stops the monitor (JES2 will restart it automatically within a few minutes)

```
$jstop
$HASP9101 MONITOR STOPPING
$HASP9085 JES2 MONITOR ADDRESS SPACE STOPPED FOR JES2
IEF404I IEESYSAS - ENDED - TIME=00.31.08
IEF403I IEESYSAS - STARTED - TIME=00.31.34
$HASP9084 JES2 MONITOR ADDRESS SPACE STARTED FOR JES2
```

This command shuts down the monitor address space. JES2 (if it is active) will restart the address space in a few minutes. The command is intended to recycle the monitor to correct any errors it may be having or to clear any history it may be keeping. Recycling the monitor will NOT pick up any new fixes to the monitor code. If a fix needs to be applied to the monitor, the JES2 address space must be recycled (a hot start will work) Note that JES2 will also restart the monitor address space if it's cancelled, forced, CALLRTM'ed, or otherwise vaporized.

$J D STATUS

$J D STATUS

- *jesx*MON.DISPLAY.STATUS (READ)
- Displays current status of JES2
- Use to identify potential JES2 problems

```
$jdstatus
$HASP9120 D STATUS
$HASP9121 OUTSTANDING ALERTS
$HASP9211 JES2 MAIN TASK NOT RUNNING
DURATION-000:00:28.14
$HASP9150 JES2 NOTICES
$HASP9159 JES2 EXECUTION PROCESSING STOPPED ($PXEQ)

$jdstatus
$HASP9120 D STATUS
$HASP9121 NO OUTSTANDING ALERTS
$HASP9150 NO JES2 NOTICES
```

This is the primary command to determine what problems may exist in JES2. Only conditions that the monitor considers potential problems are displayed. Two types of information are displayed:

Alerts - these are the alerts for which the monitor has already issued a message
Notices - these are other conditions which can exist in JES2 which could be contributing to a problem, but are not things the monitor surfaces via alerts.

$J D JES

$J D JES (or **$J D HASP**)

- *jesx*MON.DISPLAY.JES (READ)
- Displays information about JES2
- Displays events that are being tracked but are not necessarily problems

```
$jdjes
$HASP9120 D JES
$HASP9121 NO OUTSTANDING ALERTS
$HASP9122 INCIDENTS BEING TRACKED
$HASP9204 JES2 MAIN TASK BUSY
DURATION-000:00:09.55 PCE-COMM     EXIT-NONE JOB ID-NONE
COMMAND-$TMONITOR,WAIT2
$HASP9203 LONG PCE DISPATCH
DURATION-000:00:09.55 PCE-COMM     EXIT-NONE JOB ID-NONE
COMMAND-$TMONITOR,WAIT2
$HASP9201 JES2 MAIN TASK WAIT DETECTED AT 7F6FC5B6
DURATION-000:00:09.51 PCE-COMM     EXIT-NONE JOB ID-NONE
COMMAND-$TMONITOR,WAIT2
$HASP9207 JES2 CHECKPOINT LOCK HELD
DURATION-000:00:09.75
$HASP9150 JES2 NOTICES
$HASP9158 JES2 PROCESSING STOPPED, $S NEEDED
```

This command is very similar to $JDSTATUS. The major difference is that this displays information that may not be a problem in addition to what the monitor considers a problem. Notice there are 4 items being tracked at the time of this command. Any one of these could, if it lasts long enough, become an alert.

Notice Messages

- Notice information

```
$HASP9150 {NO} JES2 NOTICES
$HASP9151 JES2 ADDRESS SPACE NOT ACTIVE
$HASP9152 JES2 INITIALIZING
$HASP9153 JES2 TERMINATING
$HASP9154 CKPT RECONFIGURATION IN PROGRESS
$HASP9155 ADDRESS SPACES WAITING FOR INTERNAL READERS
$HASP9156 ADDRESS SPACES WAITING FOR SPOOL SPACE
$HASP9157 CANNOT RESTART JES2, IPL REQUIRED
$HASP9158 JES2 PROCESSING STOPPED, $S NEEDED
$HASP9159 JES2 EXECUTION PROCESSING STOPPED ($PXEQ)
$HASP9160 AT LEAST ONE PCE HAS ENDED
$HASP9161 NOT ALL SPOOL VOLUMES ARE AVAILABLE
$HASP9162 PCES WAITING FOR SPOOL SPACE
$HASP9163 FAST SPOOL GARBAGE COLLECTION (SPOOLDEF GCRATE=FAST)
```

Notices are conditions that arise in JES2 but are not time related in nature (it does not matter how long the condition existed). Notices are displayed on the $JDJES and $JDSTATUS commands. Many have related JES2 messages or commands that further explain the situation. They are gathered here to provide a single place to determine what is happening in JES2.

$J D DETAILS

$J D DETAILS

- *jesx*MON.DISPLAY.DETAILS (READ)
- Displays detailed information about JES2 resources, sampling and MVS waits
- Intended as a diagnostic aid

```
$jddetails
$HASP9103 D DETAIL
$HASP9104 JES2 RESOURCE USAGE SINCE 2004.034 19:00:00
RESOURCE     LIMIT    USAGE      LOW     HIGH  AVERAGE
-------- --------- -------- -------- -------- --------
BERT          1100       91       91       91       91
BSCB            42        0        0        0        0
BUFX           249        0        0        0        0
CKVR             2        0        0        0        0
CMBS           208        0        0        2        1
CMDS           200        0        0        1        0
ICES           101        0        0        0        0
JNUM          9999        7        7        7        7
JOES           200        3        3        3        3
JQES           500        7        7        7        7
LBUF           151        0        0        0        0
NHBS            23        0        0        0        0
SMFB            53        0        0        0        0
TGS            525      243      243      243      243
TTAB             3        0        0        0        0
VTMB            24        0        0        0        0
```

This command displays various information that the monitor is collecting..

$HASP9104 displays resource usage - currently for all of the resources that are tracked by the $HASP050 message in the JES2 main task. The intent is to get all of the information in a single display and have it available when JES2 commands can't be processed (for example, if a serious CMB shortage exists).

The resource usage is reset at the top of every hour (low, high, and average). Previous usage can be displayed by the $J D HISTORY command.

$J D DETAILS *(continued)*

$J D DETAILS continued

```
$HASP9105 JES2 SAMPLING STATISTICS SINCE 2004.034 19:00:00
TYPE               COUNT  PERCENT
----------------  ------  -------
ACTIVE              3372    11.92
IDLE               23850    84.36
LOCAL LOCK             0     0.00
NON-DISPATCHABLE       0     0.00
PAGING                 0     0.00
OTHER WAITS         1048     3.70
TOTAL SAMPLES      28270
$HASP9106 JES2 MAIN TASK MVS WAIT TABLE
DATE     TIME     ADDRESS  MODULE   OFFSET WT-COUNT SM-COUNT PCE XIT
-------- -------- -------- -------- ------ -------- -------- --- ---
2004.034 18:47:00 7F6FC5B6 UNKNOWN +000000        1     1048  10 JCO
2004.034 18:46:28 010E6D80 IGC018  +000B40        4       10  23 JCO
2004.034 18:46:25 0003D5B2 HASPCKPT+0065B2        1        2  23 JCO
2004.034 18:46:25 0003E25A HASPCKPT+00725A        1        1  23 JCO
2004.034 18:46:26 06B057D2 HASPIRDA+0027D2        2       36  23 JCO
2004.034 18:46:28 0125767E IEWFETCH+00152E        1        1  23 JCO
2004.034 18:46:28 00066FFA HASPDYN +000FFA        1        1  23 JCO
$HASP9107 JES2 ERROR COUNTS SINCE 2004.034 19:00:00
ERR-TYPE    COUNT
-------- --------
MAIN            1
```

New in z/OS 1.5!

Sampling statistics display counts and percentages of what the sampler detected the JES2 main task was doing. It too is reset at the top of every hour.
The main task wait table is maintained until the monitor is recycles. It has information on explicit waits of the JES2 main task. A wait of the main task could indicate a potential problem. The WT-COUNT is the number of unique times the sampler encountered a wait. The SM-COUNT is the number of time the sampler saw the wait. So in this case, the first wait was encountered once and sampled 1048 times. Since we sample 20 times a second, this wait lasted about 52 seconds. If the WT-COUNT was 2, then that would imply that we waited at the wait 2 times for an average of 26 seconds each time.
The PCE and XIT column gives information on what PCE and exit were in control at the time of the wait. PCE can be a number (the decimal PCE id) or MLT if multiple PCE types waited. XIT can be an exit number, MLT (multiple exits), JCO (JES2 Code Only), or JNX (JES2 and exits).
In z/OS 1.5, information about disastrous and catastrophic errors is also displayed. Any non-zero counts for the following types of errors will be displayed:

- MAIN - JES2 main task ABEND or $ERROR
- DISTERR - disastrous errors ($HASP096)
- CBIO - $CBIO errors (CBIMPLx $DISTERRs)
- SUBTASK - JES2 subtask ABENDs
- OTHER - PQE, CKPT, and other errors

$J D DETAILS *(continued)*

$J D DETAILS continued

- As of APAR OA06186, a subscript is allowed
  - **$JDDETAILS(RESOURCE)** - resource data only
  - **$JDDETAILS(MAIN)** - sampling statistics only
  - **$JDDETAILS(WAIT)** - MVS WAIT information only
  - **$JDDETAILS(ERROR)** - error information only in **z/OS 1.5**

APAR OA06186 allows the amount of output from the $JDDETAILS to be limited to just one of the specific detail types: RESOURCE, MAIN, WAIT, or ERROR

$J D HISTORY

$J D HISTORY

- *jesx*MON.DISPLAY.HISTORY (READ)
- Display history information (up to 72 hours)

```
$HASP9130 D HISTORY
$HASP9131 JES2 BERT USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW     HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00     1100       91       91       91       91
2004.034 18:31:34     1100       91       91       91       91
$HASP9131 JES2 BSCB USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW     HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00       42        0        0        0        0
2004.034 18:31:34       42        0        0        0        0
$HASP9131 JES2 BUFX USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW     HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00      249        0        0        0        0
2004.034 18:31:34      249        0        0        0        0
$HASP9131 JES2 CKVR USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW     HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00        2        0        0        0        0
2004.034 18:31:34        2        0        0        0        0
$HASP9131 JES2 CMBS USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW     HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00      208        0        0        0        0
2004.034 18:31:34      208        0        0        2        0
```

The resource usage and sampling statistics displayed by the $JDDETAILS command are reset every hour. Previous values are retained and displayed on the $JDHISTORY command. Up to 72 hours of history are displayed.

$J D HISTORY
(continued)

$J D HISTORY Continued

```
$HASP9131 JES2 CMDS USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00       200        0        0        0        0
2004.034 18:31:34       200        0        0        1        0
$HASP9131 JES2 ICES USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00       101        0        0        0        0
2004.034 18:31:34       101        0        0        0        0
$HASP9131 JES2 JNUM USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00      9999        7        7        7        7
2004.034 18:31:34      9999        7        7        7        7
$HASP9131 JES2 JOES USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00       200        3        3        3        3
2004.034 18:31:34       200        3        3        3        3
$HASP9131 JES2 JQES USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00       500        7        7        7        7
2004.034 18:31:34       500        7        7        7        7
$HASP9131 JES2 LBUF USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00       151        0        0        0        0
2004.034 18:31:34       151        0        0        0        0
```

$J D HISTORY
(continued)

$J D HISTORY Continued

```
$HASP9131 JES2 NHBS USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00        23        0        0        0        0
2004.034 18:31:34        23        0        0        0        0
$HASP9131 JES2 SMFB USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00        53        0        0        0        0
2004.034 18:31:34        53        0        0        0        0
$HASP9131 JES2 TGS  USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00       525      243      243      243      243
2004.034 18:31:34       525      243      243      243      243
$HASP9131 JES2 TTAB USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00         3        0        0        0        0
2004.034 18:31:34         3        0        0        0        0
$HASP9131 JES2 VTMB USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00        24        0        0        0        0
2004.034 18:31:34        24        0        0        0        0
```

$J D HISTORY
(continued)

$J D HISTORY Continued

```
$HASP9132 MAIN TASK SAMPLING PERCENT HISTORY
DATE     TIME        COUNT ACTIVE    IDLE    WAIT L-LOCK N-DISP PAGING
-------- --------  ------ ------ ------ ------ ------ ------ ------
2002.186 19:00:00   14677   0.16  99.83   0.00   0.00   0.00   0.00
2002.186 18:31:34   33660  10.04  86.84   3.11   0.00   0.00   0.00
$HASP9133 JES2 ERROR HISTORY
DATE     TIME          MAIN  DISTERR     CBIO  SUBTASK    OTHER
-------- -------- -------- -------- -------- -------- --------
2004.034 19:00:00        0        0        0        0        0
2004.034 18:31:34        1        0        0        0        0
```

➢ New in z/OS 1.5!

$J D HISTORY
(continued)

$J D HISTORY

Excerpt from a longer running system

```
.
.
.
$HASP9131 JES2 TGS  USAGE HISTORY
DATE     TIME         LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.035 14:00:00      3525     1183     1002     1312     1143
2004.035 13:00:00      3525     1157      907     1351     1106
2004.035 11:00:00      3525      952      936     1406     1182
2004.035 10:00:00      3525     1129      962     1437     1198
2004.035  9:00:00      3525     1105      932     1443     1198
2004.035  8:00:00      3525     1302     1061     1596     1307
2004.035  7:00:00      3525     1161      962     1525     1266
2004.035  6:00:00      3525     1339      991     1646     1253
2004.035  5:00:00      3525     1285      978     1564     1293
2004.035  4:00:00      3525     1399      974     1530     1266
2004.035  3:00:00      3525     1378     1094     1656     1346
2004.035  2:00:00      3525     1343     1102     1856     1476
2004.035  1:00:00      3525     1505     1071     1875     1400
2004.035  0:00:00      3525     1787     1280     1958     1562
2004.034 23:00:00      3525     1880     1320     2039     1676
2004.034 22:00:00      3525     1718     1424     2325     1849
2004.034 21:00:00      3525     1869     1390     2373     1837
2004.034 20:10:50      3525     1975     1438     2253     1836
.
.
.
```

This is an excerpt from a $J D HISTORY command on a system that has been running for several hours. For each entry, there is a line for every hour of activity. Since up to the last 72 hours of activity are tracked, the output can be very long.

$J D HISTORY
(continued)

$J D HISTORY

Excerpt from a longer running system

```
.
.
.
$HASP9132 MAIN TASK SAMPLING PERCENT HISTORY
DATE     TIME        COUNT ACTIVE   IDLE   WAIT L-LOCK N-DISP PAGING
-------- --------    ------ ------ ------ ------ ------ ------ ------
2004.035 14:00:00    15540   3.64  96.26   0.00   0.09   0.00   0.00
2004.035 13:00:00    69286   2.84  97.07   0.00   0.07   0.00   0.00
2004.035 11:00:00    70393   3.12  96.79   0.00   0.07   0.00   0.00
2004.035 10:00:00    70577   2.61  97.33   0.00   0.05   0.00   0.00
2004.035  9:00:00    69129   3.59  96.32   0.00   0.08   0.00   0.00
2004.035  8:00:00    70234   2.87  97.06   0.00   0.05   0.00   0.00
2004.035  7:00:00    70834   2.68  97.26   0.00   0.04   0.00   0.00
2004.035  6:00:00    70848   2.75  97.19   0.00   0.04   0.00   0.00
2004.035  5:00:00    70510   2.69  97.24   0.00   0.05   0.00   0.00
2004.035  4:00:00    70876   2.72  97.21   0.00   0.05   0.00   0.00
2004.035  3:00:00    70979   2.99  96.94   0.00   0.05   0.00   0.00
2004.035  2:00:00    71020   2.96  96.98   0.00   0.05   0.00   0.00
2004.035  1:00:00    70960   3.26  96.68   0.00   0.05   0.00   0.00
2004.035  0:00:00    71060   3.33  96.60   0.00   0.05   0.00   0.00
2004.034 23:00:00    71020   3.77  96.14   0.00   0.07   0.00   0.00
2004.034 22:00:00    71000   4.13  95.77   0.00   0.08   0.00   0.00
2004.034 21:00:00    71038   4.40  95.48   0.00   0.10   0.00   0.00
2004.034 20:10:50    58174   4.48  95.41   0.00   0.09   0.00   0.00
.
.
.
```

Note that in this case, the number of samples for the periods representing a full hour range from 69129 to 71060, which averages 19.2 to 19.7 samples per second. The tracking of probes and alerts are based on "sample seconds", which equates to 20 samples rather than wall clock seconds. This is primarily to deal with test systems, especially second level VM systems which may be stopped for long periods of time. However, all times reported in messages represent actual wall clock times.

$J D HISTORY *(continued)*

- **OA06186** allows $JDHISTORY output to be limited to a subset of resources and a subset of time
  - Subscripts limit which resource(s) are displayed
    - *resource type*, **MAIN**, **ERROR** (z/OS 1.5 only)
  - **HOURS=** parameter indicates how many intervals to display

```
          $jdhistory(jqes,joes),hours=2
          $HASP9130 D HISTORY
$HASP9131 JES2 JQES     USAGE HISTORY
DATE     TIME        LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 22:00:00      500       26       25       29       26
2004.034 21:00:00      500       26       18       28       26
$HASP9131 JES2 JOES     USAGE HISTORY
DATE     TIME        LIMIT    USAGE      LOW      HIGH  AVERAGE
-------- -------- -------- -------- -------- -------- --------
2004.034 22:00:00      200       19       19       19       19
2004.034 21:00:00      200       19       15       19       17
```

The output of $JDHISTORY can be quite voluminous if many hours of sampling data are available (72 hours by 16+ resources)
OA06186 addresses this by allowing the amount of output to be limited to a subset of resources, and a subset of sampling intervals.

Subscripts can be specified to limit which resources are displayed, and can include one or more of the following: BERT, BSCB, BUFX, CKVR, CMDS, ICES, JNUM, JOES, JQES, LBUF, NHBS, SMFB, TGS, TTAB, VTMB, MAIN, ERROR

The HOURS= parameter limits the number of lines displayed for each resource. The most recent intervals will be displayed.

Other goodies

- Dumps including the JES2 address space also include the monitor address space
- Dumps including the monitor address space also include the JES2 address space
- IPCS formatting of JES2 monitor control blocks or sampling data in z/OS 1.5
- Main task save areas ($PSV) now include a time stamp (**PSVSTCK**)

PERFDATA

- The $D PERFDATA command can provide interesting information in conjunction with monitor commands
  - **$D PERFDATA(QSUSE)** displays long waits for CKPT
  - **$D PERFDATA(CPUSTAT)** displays percentage of total JES2 CPU used by each specific PCE type
  - **$D PERFDATA(PCESTAT)** displays what resources PCEs are $WAITing for, and for how long
  - **$D PERFDATA(EVENT)** tracks, among other things, long PCE dispatches
- Note: The JES2 address space must be responding to commands for this to be useful

$D PERFDATA Example 1

- Alert

```
19.23.39 $HASP9201 JES2 MAIN TASK WAIT DETECTED AT HASTDIAG+01FDEE
DURATION-000:00:13.51 PCE-COMM     EXIT-NONE JOB ID-NONE
COMMAND-$TMONITOR,WAIT1
```

- $J D DETAILS

```
...
$HASP9106 JES2 MAIN TASK MVS WAIT TABLE
DATE     TIME     ADDRESS  MODULE   OFFSET WT-COUNT SM-COUNT PCE XIT
-------- -------- -------- -------- ------ -------- -------- --- ---
2002.197 19:23:02 00066FFA HASPDYN +000FFA        2        2  23 JCO
2002.197 19:23:02 0125767E IEWFETCH+00152E        3        3  23 JCO
2002.197 19:23:02 0003FBC0 HASPCKPT+008BC0        1        6  23 JCO
2002.197 19:23:25 070E1DEE HASTDIAG+01FDEE        1      471  10 JCO
```

- $D PERFDATA(EVENT)

```
$HASP660 $DPERFDATA(EVENT)
$HASP660 EVENT INFORMATION - INTERVAL=1:22.170285,
$HASP660 TIME=2002.197,19:23:49.61,EVENT=LONG PCE DISPATCH,PCE=COMM,
$HASP660 MOD=HASPCOMM,01990500,DATA=$TMONITO,DURATION=23.957838
```

Here's an example

In this case, JES2 has gone into a long MVS WAIT.

While JES2 is waiting, an ALERT will be issued showing that JES2 is in an MVS WAIT.

The $J D DETAILS command shows a list of all unexpected MVS WAITs. This wait shows up on the list, and will remain in the list even after the WAIT has been resolved and the ALERT has been cleared.

The $D PERFDATA(EVENT) will also notice that a long time passed between JES2 dispatcher calls. There's advantages and disadvantages to this command:

Disadvantages: First of all, the command obviously won't work while JES2 is stuck in the WAIT. Also, it's only identifiable as a long PCE dispatch, not a loop near HASTDIAG+1FDEE as the $J D DETAILS command does.

Advantages: $D PERFDATA will track all long PCE dispatches, not just MVS WAITs. So, had this been a loop, the only indication once the loop has terminated is the $D PERFDATA command.

$D PERFDATA Example 2

- $J D DETAILS

```
...
$HASP9105 JES2 SAMPLING STATISTICS SINCE 2002.197 20:00:00
TYPE               COUNT  PERCENT
----------------  ------  -------
ACTIVE             17036    97.64
IDLE                 411     2.35
LOCAL LOCK             0     0.00
NON-DISPATCHABLE       0     0.00
PAGING                 0     0.00
OTHER WAITS            0     0.00
TOTAL SAMPLES      17447
...
```

- $D PERFDATA(CPUSTAT)

```
20.20.18  $HASP660 $DPERFDATA(CPUSTAT)
$HASP660 CPU PERFORMANCE STATISTICS - INTERVAL=25:15.871322,CPU=
$HASP660 23:08.826855,
...
$HASP660 PCENAME=COMM,CPU%=99.82,CPU=23:06.436990,TIME=24:53.583293,
$HASP660 QSUSE_TIME=0.000000,IOCOUNT=0,CKPT_COUNT=0,
...
```

Here's another example

In this case, JES2 is very unresponsive and slow, and appears to be using a lot of CPU, but no monitor ALERTs are being issued.

$J D DETAILS confirms that JES2 is using massive amounts of CPU, but doesn't indicate where

$D PERFDATA(PCESTAT) displays, for every PCE type, what percentage of the CPU time spent in JES2 is in that type (all percentages add up to 100%). In this case, we see over 99% in COMM. Additional information about the COMM PCE from this display, like where $WAITs were done, can be useful in isolating exactly which commands are involved

$D PERFDATA Example 2 *(continued)*

- **$J D JES**

```
20.38.31  $jdjes
20.38.31  $HASP9120 D JES
$HASP9121 NO OUTSTANDING ALERTS
$HASP9122 INCIDENTS BEING TRACKED
$HASP9204 JES2 MAIN TASK BUSY
DURATION-000:00:06.35 PCE-COMM     EXIT-NONE JOB ID-NONE
COMMAND-$TMONITOR,LOOP1=1
$HASP9203 LONG PCE DISPATCH
DURATION-000:00:06.36 PCE-COMM     EXIT-NONE JOB ID-NONE
COMMAND-$TMONITOR,LOOP1=1
$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR HASTDIAG+01FC4E
DURATION-000:00:06.28 PCE-COMM     EXIT-NONE JOB ID-NONE
COMMAND-$TMONITOR,LOOP1=1
$HASP9150 NO JES2 NOTICES
```

- **$D PERFDATA(EVENT)**

```
...
$HASP660 TIME=2002.197,20:48:43.99,EVENT=LONG PCE DISPATCH,PCE=COMM,
$HASP660 MOD=HASPNUC,97330800,DATA=$TMONITO,DURATION=6.124284
$HASP660 TIME=2002.197,20:48:50.95,EVENT=LONG PCE DISPATCH,PCE=COMM,
$HASP660 MOD=HASPNUC,97330800,DATA=$TMONITO,DURATION=6.954167
$HASP660 TIME=2002.197,20:48:57.32,EVENT=LONG PCE DISPATCH,PCE=COMM,
$HASP660 MOD=HASPNUC,97330800,DATA=$TMONITO,DURATION=6.362127
...
```

Just to see what's going on, issue a $J D JES command. In this case, we get lucky and see that the monitor is tracking a long PCE dispatch and a potential loop that's been going on for a few seconds but not long enough to issue an alert.
$D PERFDATA(EVENT) shows that lots of 6-7 second long dispatches are occurring, one after another.
There are two things worth noting in this example

- If the loops we see here had terminated in less than 5 seconds, neither of these two commands would have tracked anything
- In the first 5 seconds of each event, $J D JES would not yet be tracking an incident. So, if it doesn't display anything, it can't hurt to issue it more than once.

Summary

- The JES2 monitor is NOT intended as a complete performance monitor (yet?)
  - Only detects extreme conditions, and displays basic performance data when JES2 cannot do so.
- Monitor messages, by themselves, do not necessarily indicate a problem exists
  - Use to detect or confirm the existence of other problems
  - ***JES2 Messages*** lists further actions to confirm problem
- Your mileage may vary
  - Understanding what's normal can help identify problem area when system is sick

Here's a sample from JES2 Messages:

$HASP9202 POTENTIAL JES2 MAIN TASK LOOP DETECTED NEAR *module+offset*
DURATION-*hh:mm:ss.xx* **PCE-***pcename* **EXIT-***xit* **JOB ID-***jobid*
COMMAND-*jes2_command*

Explanation: This is an alert message which is issued in response to a $JDJES or a $JDSTATUS JES2 monitor command or as a highlighted message. Based on context, this message may indicate an alert or a incident being tracked. This message indicates a potential loop condition exists in the JES2 main task. Loops may indicate a problem or complex processing that takes a long time to complete. One way to determine if the loop is making progress is the job information in the message. If it is changing over time then the loop may eventually complete. If it is not changing, it may indicate a problem with the processing or the current job. In general, most functions in JES2 processes jobs in job number order. Warm start processes jobs in job index order.

System Action: If condition persists, this message is re-issued every 30 seconds with updated information.

Operator response: Monitor alerts indicate potential reasons why JES2 is not functioning properly. If this condition persists and is impacting normal JES2 operations, notify the system programmer.

System Programmer response: Loops can be an indication of a tuning problem, a complex process that takes a long time to complete or an error situation. If the JES2 main task is not getting enough CPU resource, it could appear to the monitor as a loop. Use the D A,JES2 command D A,JES2 command or a performance monitor to ensure that JES2 is indeed consuming CPU. Once CPU performance problems have been eliminated, examine the job information displayed in the messages to determine if it is being updated or not.

If no progress is being made JES2 needs to be ABENDed and restarted. First capture a dump of the JES2 address space using an MVS DUMP command. While JES2 is taking a dump, the $HASP9202 message may be deleted and another alert message issued. Once the dump has completed, a $PJES2,ABEND command should be entered followed by a $PJES2,ABEND,FORCE. The $PJES2,ABEND,FORCE is needed because the main task is in a loop. Once JES2 has terminated, it can be restarted. The dump should be analyzed to determine what code is looping and the cause of the loop. If the problem is in exit code (or exit invoked code) correct the exit logic. If the problem is caused by base code, contact the IBM support center.