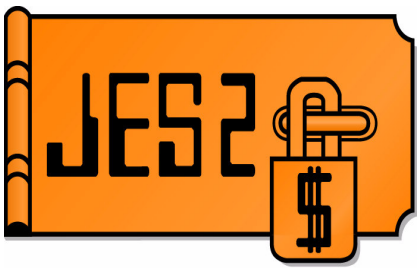
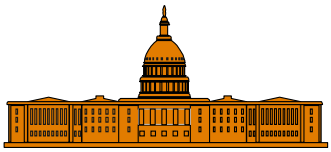


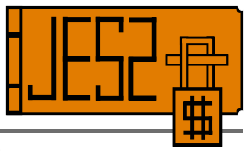
JES2 Project
Sessions 2663 and 2664
Bob Jinkins
IBM JES2 Level2

JES2 Debugging

SHARE 101, Summer 2003
Washington, DC



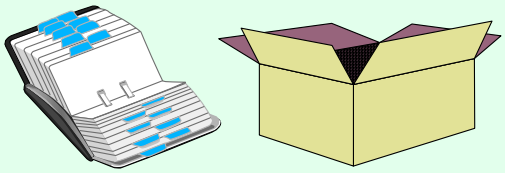
Agenda



An effective debugger has a blend of product knowledge and debugging technique. This presentation blends a debugger's view of JES2 structure and using IPCS and other tools to debug JES2.

This isn't a step-by-step debugging cookbook ... that can be written when the last bug is found. This is a source of product and tools knowledge to leverage and increase your debugging skills. Familiarity with IPCS, assembler coding, and JES2 externals is assumed.

JES2 Product Information and Packaging



COMPIDs for JES2 and related software

Component ID	Component name
5752SC1BH	JES2
5752SCJSC	JESXCF
5752SC141	JES Common
5752SC1B2	External Writer
5785BAZ00	JES/328x
5785BAC00	JES/328x
5752SC1BL	Multi-leaving Workstation
566548801	SDSF

JES2 SMP libraries

DISTLIB	SYSLIB	Description
AHASPNL0	SHASPNL0	JES2 IPCS panels
AHASMOD	SHASLINK	JES2 modules
AHASMOD	SHASMIG	JES2 IPCS modules
AHASSRC	SHASSRC	JES2 source
AHASMAC	SHASMAC	JES2 macros
AHASPARM	SHASPARM	Parmlib member (HASLIPCS)
AHASSAMP	SHASSAMP	Samples
AHASMENU	SHASMENU	U.S. English msg skeletons
AHASMJPN	SHASMJPN	Japanese msg skeletons

Naming conventions for JES2 parts

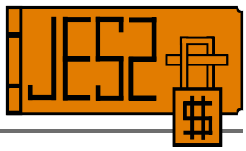
$xxxxxxx	**JES2 data areas and executable macros**

- **$** typically omitted from eyecatchers and DSECT names
- For example, DSECT name and eyecatcher for **$MIT** is **MIT**

HASPxxxx	**JES2 private area modules**
HASCxxxx	**JES2 common area modules**
HASJxxxx	**JES2 monitor modules** ⇦ **New in z/OS 1.4**
HASMxxxx	**JES2 IPCS modules**
HASLxxxx	**JES2 IPCS parmlib member and panels**
HASIxxxx	**Sample JES2 initialization statements, JES2 proc, etc. (not supported)**
HASXxxxx	**Sample JES2 exits (not supported)**
xxxxxxxx	**Anything goes for installation exits**

Terminology: "module" = part (usually single CSECT) vs. "load module"

JES2 load modules



HASJES20 **Main JES2 private area load module**
- Contains HASPxxxx modules beginning with HASPNUC

HASPINIT **JES2 initialization load module**
- Contains HASPIRxx modules beginning with HASPIRA

HASCxxxx **Common area JES2 load modules**
- One load module per HASCxxxx module (CSECT)

HASJ2MON **JES2 monitor load module** ⇐ **New in z/OS 1.4**
- Contains HASJxxxx modules beginning with HASJMON

HASMFMTM **Main JES2 IPCS load module**
- Contains HASMxxxx modules beginning with HASMFMTM

HASMWTLB **JES2 IPCS MVS SYSLOG buffer display**
- Contains HASMxxxx modules beginning with HASMWTLB

HASMxxxx **Other JES2 IPCS load modules**
- One load module per HASMxxxx module (CSECT)

xxxxxxxx **Anything goes for installation exits**

Warning about JES2 control blocks

Don't rely on JES2 data area books

- **JES2 control block field names, offsets, lengths, and usage can change in new releases and in the service stream**
  - JES2 control blocks ***don't*** change compatibly
  - Good for innovation ... inconvenient for debugging
  - ☆ Make sure JES2 IPCS level corresponds with JES2 level in dump
- ***Never*** hard-code offsets
- Reassemble **exits**, **SDSF**, and applicable **ISV/OEM** products ***every*** time you apply service
  - ☆ Problems frequently far removed from source and difficult to diagnose

Caution from JES2 Exits book

Defining exits and writing installation exit routines is intended to be accomplished by experienced system programmers; the reader is assumed to have knowledge of JES2.

If you want to customize JES2, IBM recommends that you use JES2 installation exits to accomplish this task. IBM does not recommend or support alteration of JES2 source code. If you assume the risk of modifying JES2, then also assure your modifications do not impact JES2 serviceability using IPCS. Otherwise, LEVEL2 may not be able to read JES2 dumps taken for problems unrelated to the modifications.

Avoid expanding JES2 control blocks. Use alternatives such as:

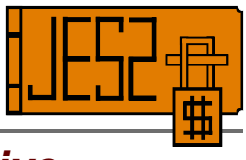
1. **Use fields dedicated for installation use that appear in many major control blocks. Place your data, or a pointer to your data, in these fields. However, beware of setting storage address in checkpointed or SPOOL resident control blocks.**
2. **Use $JCTX services rather than modifying $JCT.**
3. **Use table pairs and dynamic tables. For example, use dynamic $BERTTABs with CBOFF=* instead of modifying $JQE.**

This is a partial list. Evaluate your specific situation and take appropriate action.

JES2 IPCS

JES2 IPCS

Installing JES2 IPCS



- ***System programmer must be proactive***
- How **IPCS datasets** map to **JES2 SMP libraries**
  - **Parmlib member** **SHASPARM** (member **HASLIPCS**)
  - **Panel library** **SHASPNL0**
  - **MIGLIB** **SHASMIG**
- External documentation
  - ***JES2 Migration*** book for install information
  - ***JES2 Diagnosis*** book for install verification procedure
- Customers have installed JES2 IPCS on the fly
  - ☆ ***Most common problem is old or missing* HASLIPCS *parmlib member (from* parmlib *or* IPCSPARM DD*)***
  - May need to drop and reinitialize dump if JES2 formatting previously attempted and control blocks are now truncated

Installing JES2 IPCS *(continued)*

- Install tips
  1. Put **SHASPARM** in **parmlib** concatenation
     - Remove old **HASLIPCS** member from **SYS1.PARMLIB** (and other datasets)
     - If your IPCS setup uses **IPCSPARM DD**, then include **SHASPARM** in concatenation
       - Use **IPCSPARM DD** if using IPCS on system at a different release
  2. Put **SHASPNL0** in **ISPPLIB** concatenation
  3. Put **SHASMIG** in **STEPLIB** or **LINKLIST** concatenation
- Later, you'll see a corresponding 3 step install verification procedure

JES2 IPCS panels are point-and-shoot

Enable **ISPF** point-and-shoot feature

```
  Menu  Utilities  Compilers  Options  Status  Help
--------------------------------------------------------------------------------
                           ISPF Primary Option Menu
Option ===> 0

0  Settings      Terminal and user parameters
...
```

```
  Log/List  Function keys  Colors  Environ  Workstation  Identifier  Help
--------------------------------------------------------------------------------
                                ISPF Settings
Command ===>

Options                                   Print Graphics
  Enter "/" to select option                Family printer type 2
    Command line at bottom                  Device name . . . .
  /  Panel display CUA mode                 Aspect ratio  . . . 0
  /  Long message in pop-up
    Tab to action bar choices
  /  Tab to point-and-shoot fields        General
  /  Restore TEST/TRACE options             Input field pad . . N
    Session Manager mode                    Command delimiter . ;
  /  Jump from leader dots
    Edit PRINTDS Command
  /  Always show split line
    Enable EURO sign
...
```

Try it, you'll like it!

JES2 Component Data Analysis

To invoke JES2 IPCS panels

- Go to IPCS option 2.6 (MVS component data analysis)
- Locate JES2
- Verify FMID, service level, and APAR number if any

(Verification step 1 of 3)

- Then Select JES2

Are you using the correct HASLIPCS parmlib member?

```
------------------ IPCS MVS DUMP COMPONENT DATA ANALYSIS ----------------------
OPTION ===>                                                 SCROLL ===> CSR

To display information, specify "S option name" or enter S to the left
of the option desired.  Enter ? to the left of an option to display
help regarding the component support.

S Name     Abstract
S JES2     JES2 analysis for HJE7707, service level 0
_ JES3D    JES3 analysis
_ LEDATA   Language Environment formatter
...
```

APAR level of HASLIPCS if any

JES2 Component Data Analysis *(continued)*


```
----------------------- JES2 Component Data Analysis -------------------------
OPTION ===>                                              SCROLL ===> CSR
Enter JES2 name ===> JES2

Select desired option for JES2 dump:
   1 JES2 base display
   2 JES2 job control blocks
   3 JES2 job output control blocks
   4 JES2 devices
   5 JES2 processors
   6 JES2 subtasks
   7 JES2 control blocks
   8 JES2 network control blocks
   9 JES2 MAS member data
  10 JES2 checkpoint control blocks
  11 JES2 NJE/RJE control blocks
  12 JES2 BERT control blocks

Select desired option for non-JES2 dump:
 101 Select JES2 control blocks for non-JES2 address space

Debug JES2 IPCS support:
 999 Display module information for HASMFMTM and related modules
1000 Set JES2 IPCS runtime debug options; current status is NORMAL

Enter UP and DOWN commands to scroll the list of options.
Enter END command to terminate JES2 data analysis.
```

Panel level (Verification step 2 of 3)

These panels are for
JES2 FMID: HJE7707
Service level: 0
HASLPRIM panel APAR level, if any

This is the main JES2 panel

- ✓ Always select **JES2 base display** for exception analysis
- ✓ Ensure correct **panels** and **miglib**

Miglib info (Step 3 of 3)

JES2 IPCS base display


```
*** JES2 Base Display ***

Subsystem "JES2" is in address space ASID(X'018C')
Dump for JES2 release="z/OS 1.4", Product level=34, Service level=0 (pointed
     to by SSCTSUSE); CVTPRODI=HBB7707
Maximum extended region size for "JES2" is 1,746,808K (per LDAELIM)
*** NOTICE:  CCTCSHED $STAC queue not empty

*** NOTICE:  $DOMQUE list not empty ($DOMQUE^=0 in $HCT)
*** NOTICE:  Field $USER2 in $HCT has been used
*** NOTICE:  $QSUSE is in effect (per $QSONDA bit in $STATUS in $HCT)
*** NOTICE:  Checkpoint is reserved (per $CKPTRSV bit in $STATUS in $HCT)
*** NOTICE:  $SPFTIME in checkpointed HCT indicates SPOOL volumes have been
     full at least once since the last cold start
*** WARNING:  $EVENT(s) exist (PCBEVNTF^=0 in $PERFCB)

 $HCCT: 00AD0630
    +0000  CCTVRSN.. 0A        RSV..... 00000000  000000
...
```

Always use *JES2 base display*

- Use even when JES2 address space ***not*** in dump
- Ensures dump and IPCS are for same JES2 release
- ☆ Performs ***exception analysis***
- Doesn't report every exception surfaced in other reports and vice versa

Option 999 from main JES2 panel

JES2 IPCS displayes information about itself

Formats **$MITs** in **HASMFMTM** load module from IPCS user's TSO address space. Usefull for debugging JES2 IPCS itself

```
*** JES2 IPCS Install/Service Information ***

JES2 FMID = HJE7707
JES2 release = z/OS 1.4
JES2 product level = 34
JES2 service level = 1

Use for dumps at JES2 service level:
   Minimum service level = 0
   Maximum service level = 1

Module information for HASMFMTM
 $MIT: 001907C8
    +0000  MITID.... MIT       MITNAME.. HASMFMTM  MITVRSN.. z/OS 1.4
    +0014  MITUVRSN.           MITUSER..           MITCBV... 01
    ...
```

Install verification step 3 of 3
Is MIGLIB level correct?

JES2 IPCS exception reporting convention

Messages for exceptions and errors begin with

***** NOTICE:**

- JES2 status information
- Notification when selected queues not empty
- Potentially relevant initialization statement values

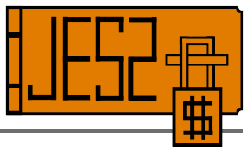
***** WARNING:**

- Unusual JES2 status
- Poor choices for initialization statement values

***** ERROR:**

- Conditions ***frequently*** indicating an error in the most common context. Use judgment, for example, a condition indicating an error during normal operation may not be an error during JES2 initialization or termination

Table driven



Much of JES2 IPCS support is table driven

- The tool consists of object-oriented table macros and a service module
- Navigates complex data structures
- Generates reports
- ☆ Extensive ***built-in analysis*** produces consistent, informative messages when control blocks cannot be found, chains broken, etc.
- ☆ Explicit analysis performed using ***analysis tables***

Consequently, there's a distinctive look and feel

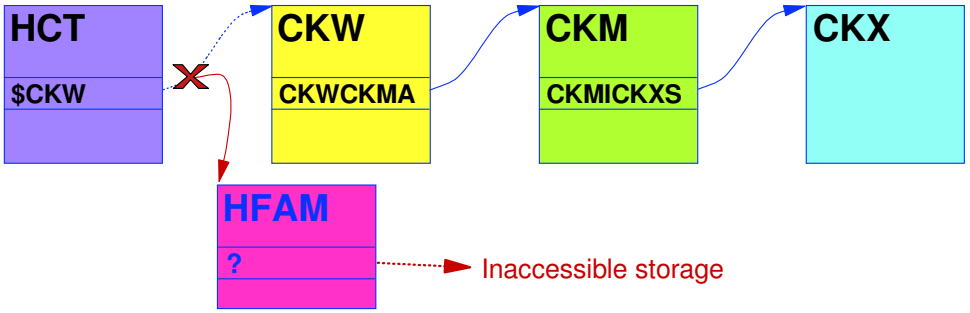
Old code still generates cryptic messages such as

UNABLE TO LOCATE CONTROL BLOCK

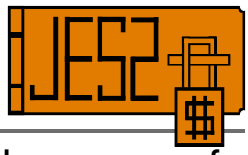
JES2 IPCS look and feel *(continued)*

Unable to find control block because of bad pointer

```
*** ERROR:  Incorrect eyecatcher of "HFAM", expected "CKW " at 1EE0F3C0 for a
     length of 0004 at offset +000000 in "$CKW" at 1EE0F3C0
*** ERROR:  Unable to access eyecatcher at 41E2D7C3 for a length of 0004 at
     offset +000000 in "$CKM" at 41E2D7C3
*** ERROR:  Unable to access field "CKMICKXS" at 41E2DBFF for a length of 0004
     at offset +00043C in "$CKM" at 41E2D7C3
*** ERROR:  Unable to locate "$CKX"
```

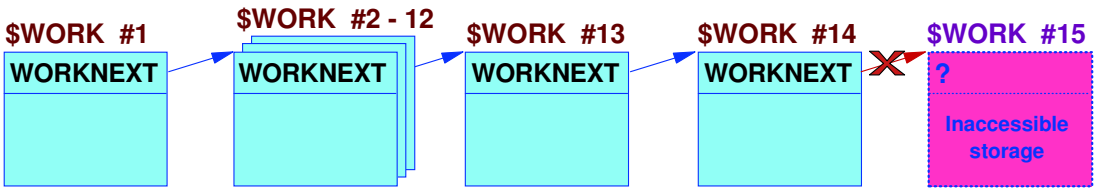


JES2 IPCS look and feel *(continued)*



Unable to continue processing linked list because of bad next pointer

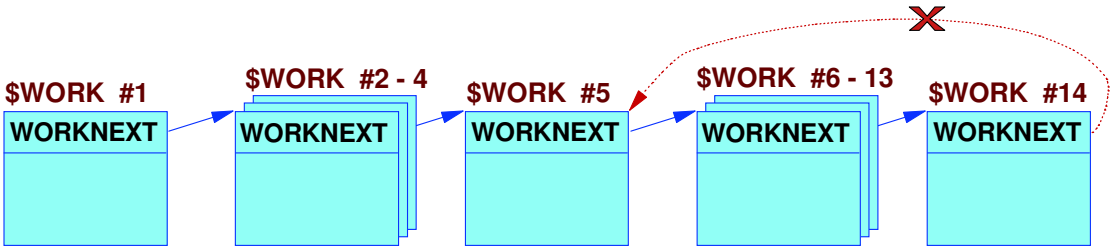
```
*** ERROR:  Unable to access field "WORKNEXT" at 00CF7E70 for a length of 0004
     at offset +000000 in "$WORK" at 00CF7E70
*** ERROR:  Unable to access forward chain field "WORKNEXT" in current "$WORK"
     at 00CF7E70, position 15 on chain.  Prior "$WORK" was at 25D68268, and
     before that at 1EE0D210.  Chain started at 25D689A0.
*** ERROR:  Unable to continue processing "available $WORK element"
```



JES2 IPCS look and feel *(continued)*

Loop detected in linked list

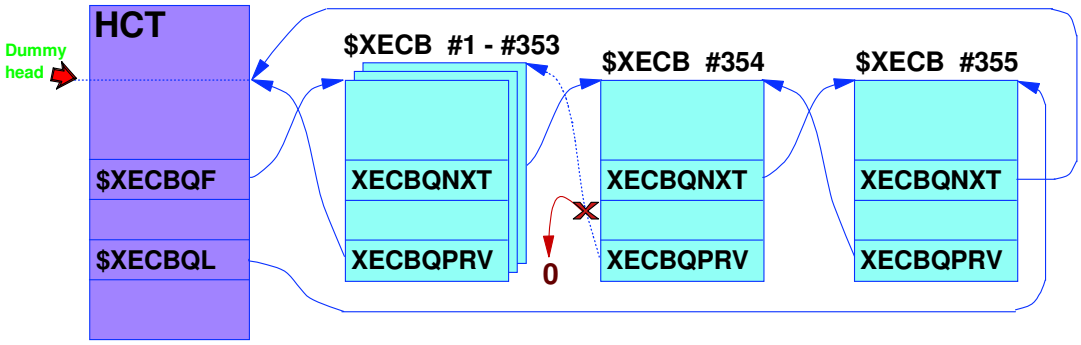
```
*** ERROR:  Loop detected in "$WORK" chain starting at 25D689A0.  Loop
     detected when chaining from "$WORK" at 25D68268 (position 14 on chain)
     to "$WORK" at 25D688F0 (position 5 on chain).
*** ERROR:  Unable to continue processing "available $WORK element"
```



JES2 IPCS look and feel *(continued)*

Bad backward pointer in doubly linked list (processing continues)

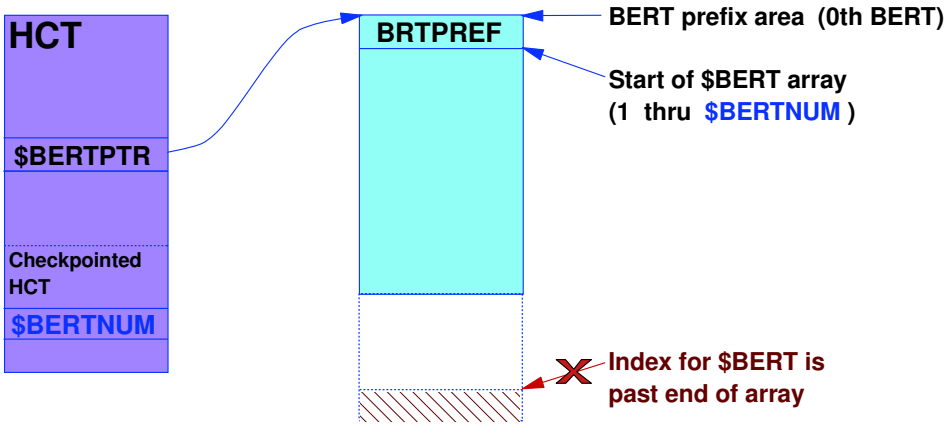
```
*** ERROR:  Backward chain field "XECBQPRV" has an incorrect value of 00000000
     at 1EE11210 at offset +000010 in next "$XECB" at 1EE11200, position 354
     on chain.  This backward chain field value does not point to current
     "$XECB" at 1E2E91E0.  Current "$XECB" was pointed to by "$XECB" at
     1E2E90D8.  Chain started at 1E2E3B40.
```



JES2 IPCS look and feel *(continued)*

Requested array subscript out of bounds

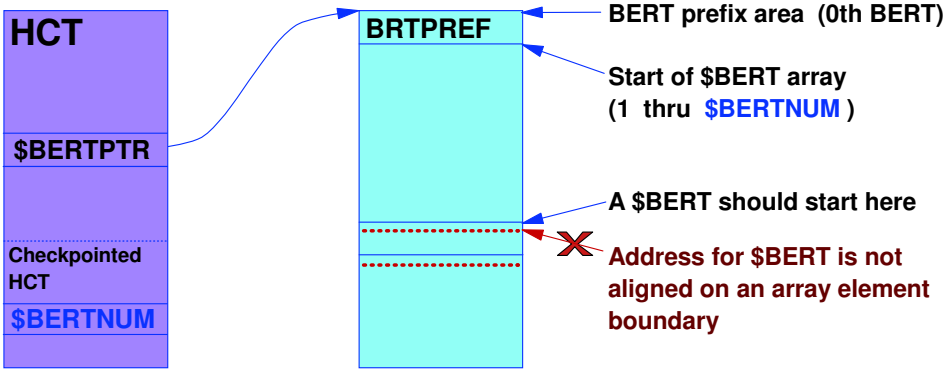
```
*** ERROR:  Array element index value of 00FF5E24 is not valid for "$BERT"
     array originating at 0F3C4018.  Valid index range is 00000001 thru
     00005E24.
*** ERROR:  Unable to continue processing "$BERT"
```



JES2 IPCS look and feel *(continued)*

Requested array element address not aligned on element boundary

```
*** ERROR:  Array element address value of 0F3FC91C is not valid for "$BERT"
     array originating at 0F3C4018.  The requested address is 00000004 bytes
     past the element at address 0F3FC918.  Valid address range is 0F3C4058
     thru 0F53C918 in increments of 00000040.
*** ERROR:  Unable to continue processing "$BERT"
```



JES2 IPCS look and feel *(continued)*

Unable to access data for a message

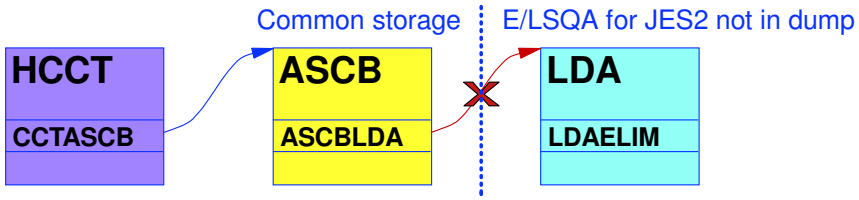
Normal message appearance

```
Maximum extended region size for "JES2" is 1,445M (per LDAELIM)
```

When message data not available

```
*** ERROR:  Unable to access eyecatcher at 7FF14EB0 for a length of 0004 at
     offset +000000 in "LDA" at 7FF14EB0 ASID(X'0017')
*** ERROR:  Unable to access field "LDAELIM" at 7FF14F88 for a length of 0004
     at offset +0000D8 in "LDA" at 7FF14EB0 ASID(X'0017')
Maximum extended region size for "JES2" is <data not available> (per LDAELIM)
```

- When you see **<data not available>**, look at prior message(s) for reason



JES2 IPCS look and feel *(continued)*

Run time error caused by IPCS install error

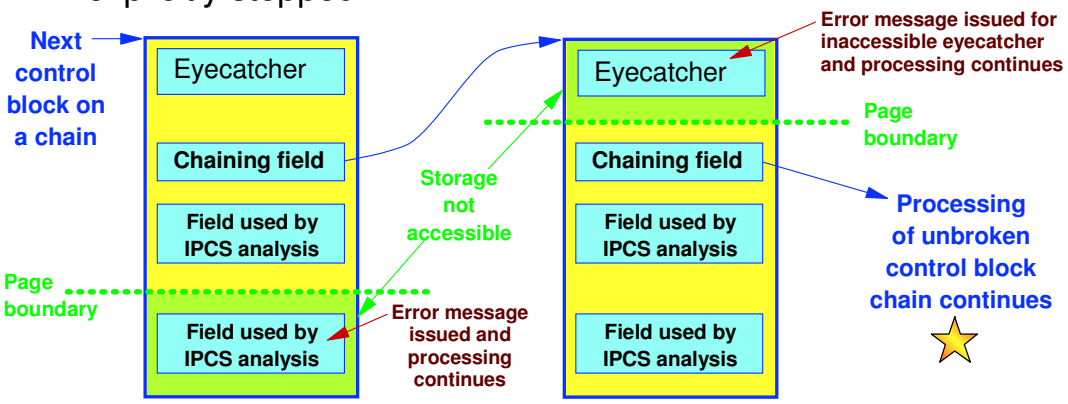
- System programmer incorrectly used up-level miglib (**SYS1.SHASMIG**) in combination with down level panel library (**SYS1.SHASPNL0**)
- While debugging a dump, JES2 IPCS tried to retrieve an undefined dialog variable from a down-level panel
- JES2 IPCS is able to continue the analysis function

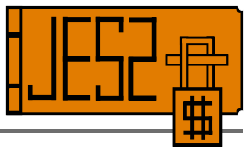
```
*** ERROR:  Unable to retrieve ISPF dialog variable "HASUSJBN" - ISPLINK
     return code 00000008
*** ERROR:  Unable to process the $SJB filter number.  The filter will not
     be used.
```

JES2 IPCS look and feel *(continued)*

Optimistic control block navigation

- Entire control blocks don't have to be accessible, only needed fields
- Eyecatcher errors reported, but processing continues
- In general processing continues whenever possible unless explicitly stopped





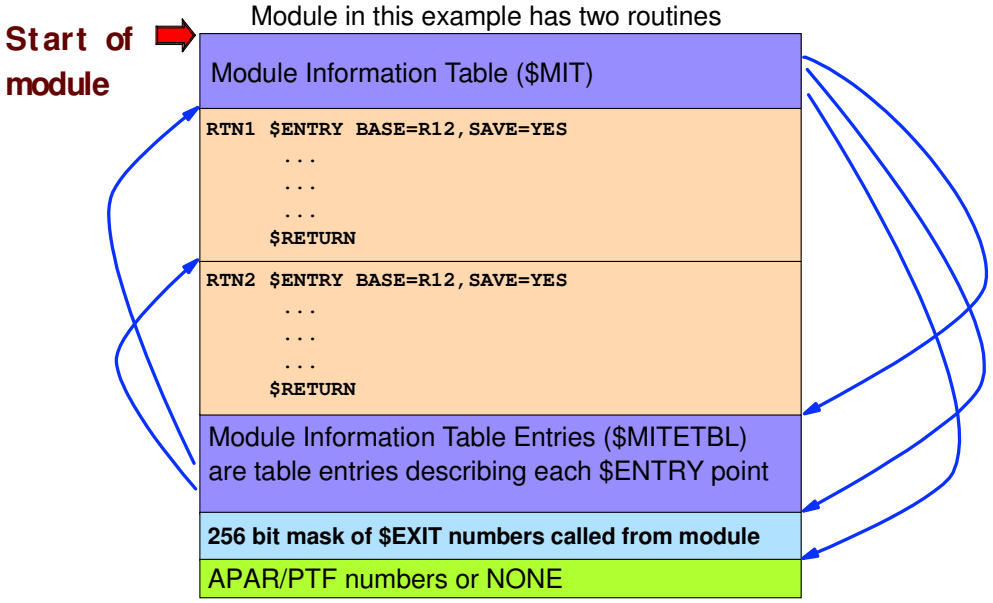
JES2 Module Layout and Displays

JES2 module layout

- JES2 module prolog conventions ***different*** from BCP
  - ***All JES2 modules reassembled for every release***
  - JES2 and installation exit modules begin with a **Module Information Table ($MIT)** data area
  - Entry points and code ***follow*** the MIT
  - More module tables follow code and data
- Service level information appears at ***end*** of module
  - Two 8 character fields contain APAR and PTF numbers or **NONE**
  - Pointed to by MIT+4C
  - Fields optional for installation exits (pointer may be zero)
  - JES2 parts contain a single CSECT
  - Installation exits may have multiple CSECTs

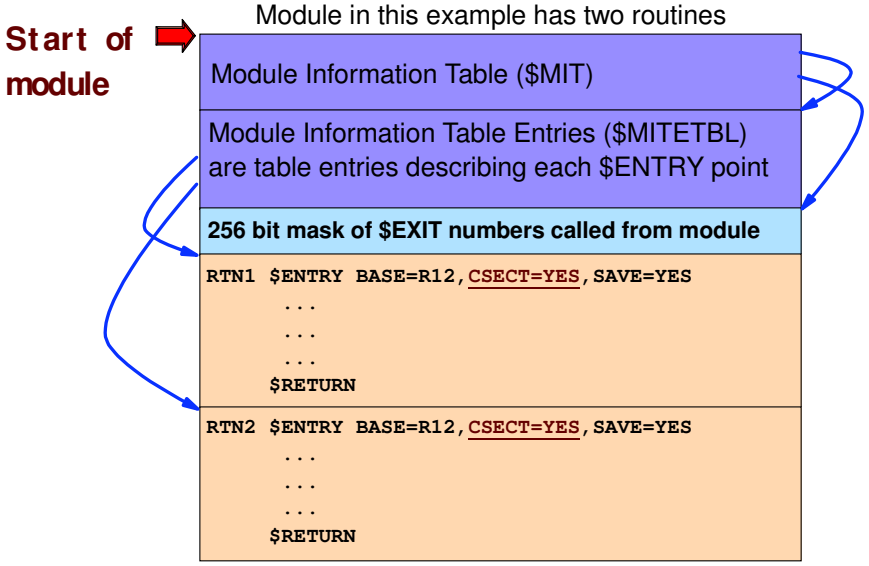
JES2 module layout *(continued)*

Typical JES2 module and installation exit module layout



JES2 module layout *(continued)*

Layout of an installation exit module using **$ENTRY** with **CSECT=YES**



JES2 module layout *(continued)*

Module Information Table ($MIT)

JES2 modules and exits start with a **MIT at offset zero**

Ignore APAR/PTF for *prior* module

MIT

```
00015440   D6E6F5F4   F9F0F940   E4E6F9F4   F5F9F940   | OW54909 UW94599  |
00015450   D4C9E340   C8C1E2D7   C3D5E5E2   A961D6E2   | MIT HASPCNVSz/OS |
00015460   40F14BF2   40404040   40404040   40404040   |  1.2             |
00015470   40404040   01E20050   F6280000   C8D1C5F7   |     .S.&6...HJE7 |
00015480   F7F0F540   F1F061F0   F761F0F2   F1F64BF1   | 705 10/07/0216.1 |
00015490   F0002838   00017BA0   00017C58   00017C78   | 0.  ...#...@...@. |
```

Module size at +41

+4C

Module name

```
00017C78                       D6E6F5F3   F9F4F840   |         OW53948  |
00017C80   E4E6F8F8   F0F5F040   D4C9E340   C8C1E2D7   | UW88050 MIT HASP |
```

- 8 character APAR and PTF numbers or **NONE** at ***end*** of module (pointed to by mit +4C)

JES2 module layout *(continued)*

Selected MIT fields

Offset	Length	Description	Example
+00	4	Eyecatcher	MIT
+04	8	Module name	HASPCNVT
+0C	8	JES2 version	z/OS 1.2
+29	1	Flags X'10000000' - Module is OCO X'01000000' - Bypass MVS macro level check X'00100000' - Module shipped with JES2 Probably exit if bit is off X'00010000' - Shipped in samplib X'00001000' - PTFNUM field exists	
+2C	8	FMID	HJE7705
+34	8	Assembly date	10/07/02
+3C	5	Assembly time	16:10
+41	3	Module length	
+44	4	Address of table of entry points ($MITETBL)	
+48	4	Address of bit mask indicating exit #'s called ($EXIT)	
+4C	4	Pointer to APAR/PTF numbers or zero	

JES2 module layout *(continued)*

Example **Module Information Table ($MIT)** display in IPCS

IP CBF 00015450 STR($MIT)

```
$MIT: 00015450
   +0000  MITID.... MIT       MITNAME.. HASPCNVS  MITVRSN.. z/OS 1.2
   +0014  MITUVRSN.           MITUSER..           MITCBV... 01
   +0025  MITENVIR. S         MITLEN... 0050      MITMVRSN. 6
   +0029  MITFLAG1. 28        RSV...... 0000      MITFMID.. HJE7705
   +0034  MITDATE.. 10/07/02  MITTIME.. 16.10     MITMODSZ. 002838
   +0044  MITENTAD. 00017BA0  MITXMAPA. 00017C58  MITAPARN. 00017C78
```

JES2 module displays

- Service applied by **SMP** may not have rolled out to production. Use JES2 commands to determine module levels, for example:

Module name

$DMODULE(HASPPRPU)

Module level

```
$HASP468 MODULE(HASPPRPU)
$HASP468 MODULE(HASPPRPU)   ADDRESS=00107778,ASSEMBLY=(09/03/02,
$HASP468                    11.22),ENVIRON=JES2,EXITPTS=(1,15),
$HASP468                    FMID=HJE7707,IBMJES2=BASE,
$HASP468                    LASTAPAR=OW53973,LASTPTF=UW89092,
$HASP468                    LENGTH=00DB40,LOADMOD=HASJES20,
$HASP468                    MACLEVEL=6,SPLEVEL=CHECK
```

Verify LENGTH when setting SLIP traps

Look for $$$$LENH near the top of the assembly

JES2 module displays *(continued)*

Specify **LONG** for routine address, etc.

$DMODULE(HASPPRPU),LONG

```
$HASP468 MODULE(HASPPRPU)
$HASP468 MODULE(HASPPRPU)  ADDRESS=00107778,ASSEMBLY=(09/03/02,
$HASP468                   11.22),ENVIRON=JES2,EXITPTS=(1,15),
$HASP468                   FMID=HJE7707,IBMJES2=BASE,
$HASP468                   LASTAPAR=OW53973,LASTPTF=UW89092,
$HASP468                   LENGTH=00DB40,LOADMOD=HASJES20,
$HASP468                   MACLEVEL=6,
$HASP468                   ROUTINES=(HASPPPI1=001077D8,
$HASP468                   PPDATE=0010CCB0,PPMSGDSP=0010CF48,
$HASP468                   PEXIT15=0010DC5A,PX15TBL=0010DD08,
$HASP468                   PRPUT=0010E5F0,PEXIT1=001131B6,
$HASP468                   HASPIMAG=00113D80,PRTASUB=00114228,
$HASP468                   PRDTCK2=00114310),SPLEVEL=CHECK,
$HASP468                   TABLES=(),VERSION=z/OS 1.4,UVERSION=
```

- **Need to set a SLIP at the beginning of a routine?**
- **No need to assemble module!**
- **Works for any routine coded with a $ENTRY**

JES2 module displays *(continued)*

To find module and address for a routine name

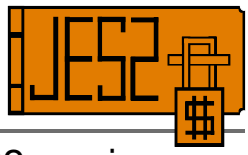
$DMODULE,ROUTINE=$DOGJQE ← **Find this routine**

```
$HASP468 MODULE(HASPJQS)
$HASP468 MODULE(HASPJQS)    ADDRESS=000A8178,ASSEMBLY=(10/24/02,
$HASP468                    12.16),ENVIRON=MIXED,EXITPTS=(14,49),
$HASP468                    FMID=HJE7707,IBMJES2=BASE,
$HASP468                    LASTAPAR=OW56117,LASTPTF=UW95072,
$HASP468                    LENGTH=00A348,LOADMOD=HASJES20,
$HASP468                    MACLEVEL=6,
$HASP468                    ROUTINES=($DOGJQE=000ADAF8),
$HASP468                    SPLEVEL=CHECK
```

Name of module containing $DOGJQE routine

Address of $DOGJQE routine

JES2 module displays *(continued)*



To find all unsupported modules loaded by JES2 services

$DMODULE,IBMJES2<>BASE,L=Z ★ Example use of filters

```
$HASP468 MODULE(EXIT3)
$HASP468 MODULE(EXIT3)      ADDRESS=0016C988,ASSEMBLY=(09/03/02,
$HASP468                    11.20),ENVIRON=JES2,EXITPTS=(),
$HASP468                    FMID=HJE7707,IBMJES2=NO,LASTAPAR=,
$HASP468                    LENGTH=000488,LOADMOD=EXIT3,
$HASP468                    MACLEVEL=6,ROUTINES=(EXIT03),
$HASP468                    SPLEVEL=CHECK,TABLES=()
$HASP468 MODULE(EXIT46)
$HASP468 MODULE(EXIT46)     ADDRESS=0016C3F0,ASSEMBLY=(09/03/02,
$HASP468                    11.20),ENVIRON=JES2,EXITPTS=(),
$HASP468                    FMID=HJE7707,IBMJES2=NO,LASTAPAR=,
$HASP468                    LENGTH=000138,LOADMOD=EXIT46,
$HASP468                    MACLEVEL=6,ROUTINES=(EXIT046),
$HASP468                    SPLEVEL=CHECK,TABLES=()
...
```

Non-IBM

★ If you want to see which exits are used, a better command to use is $DEXIT,STATUS=ENABLED

JES2 load module displays

Command to display load modules loaded by JES2 services

$DLOADMOD(HASCLINK)

```
$HASP819 LOADMOD(HASCLINK
$HASP819 LOADMOD(HASCLINK)   ADDRESS=1BDBE2A8,LENGTH=003D58,
$HASP819                     RMODE=ANY,SPLEVEL=CHECK,STORAGE=CSA
```

Info comes from $LMT control blocks

Modules loaded into CSA don't have CDEs so the IPCS WHERE command won't identify them

Finding JES2 modules in IPCS

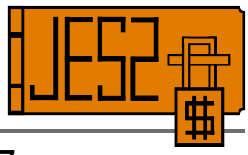
From the JES2 main panel, select option 7, then option for **$MIT** display

```
...
$MIT: 00161AF0
   +0000  MITID.... MIT       MITNAME.. HASPARMO  MITVRSN.. z/OS 1.2
   +0014  MITUVRSN.           MITUSER..           MITCBV... 01
   +0025  MITENVIR. M         MITLEN... 0050      MITMVRSN. 6
   +0029  MITFLAG1. A8        RSV...... 0000      MITFMID.. HJE7705
   +0034  MITDATE.. 05/15/01  MITTIME.. 06.44     MITMODSZ. 000288
   +0044  MITENTAD. 00161CF8  MITXMAPA. 00161D48  MITAPARN. 00161D68
   Last APAR: NONE
   Last PTF:  NONE
$MIT: 000D1290
   +0000  MITID.... MIT       MITNAME.. HASPBSC   MITVRSN.. z/OS 1.2
   +0014  MITUVRSN.           MITUSER..           MITCBV... 01
   +0025  MITENVIR. J         MITLEN... 0050      MITMVRSN. 6
   +0029  MITFLAG1. 28        RSV...... 0000      MITFMID.. HJE7705
   +0034  MITDATE.. 07/15/02  MITTIME.. 15.44     MITMODSZ. 0052B8
   +0044  MITENTAD. 000D6438  MITXMAPA. 000D6518  MITAPARN. 000D6538
   Last APAR: OW52199
   Last PTF:  UW86085
...
```

Module service level also shown

Above modules are part of load module HASJES20 so they don't show up in an IPSUMM FORMAT

Finding JES2 load modules using IPCS

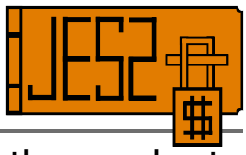


From the JES2 main panel, select option 7, then option for **$LMT** display

```
...
$LMT: 37716048
   +0000  LMTMODNM. BAJ2X002  LMTMITAD. 0016A378  LMTSUBPL. 00
   +000D  LMTMODLN. 000C88    LMTBASEA. 00000000  LMTFLG1.. 28
   +0015  LMTFLG2.. 10        RSV...... 0000      LMTCHAIN. 37716080
   +001C  RSV...... 00000000
$LMT: 37716080
   +0000  LMTMODNM. BAJ2X003  LMTMITAD. 00165B58  LMTSUBPL. 00
   +000D  LMTMODLN. 0004A8    LMTBASEA. 00000000  LMTFLG1.. 28
   +0015  LMTFLG2.. 10        RSV...... 0000      LMTCHAIN. 377160B8
   +001C  RSV...... 00000000
...
```

- **LMTMODNM - Module name (non-IBM in example based on names)**
- **LMTMITAD - Address of first $MIT**
- **LMTMODLN - Length of module**
- **See $LMT data area for flag bits**
- **Modules loaded into CSA don't have CDEs so the IPCS WHERE command won't identify them**

$MIT display

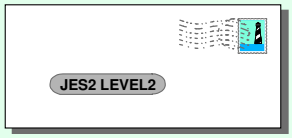


From main JES2 panel go to option 7 and then select option for $MIT

```
$MIT: 00065F30
   +0000  MITID.... MIT       MITNAME.. HASPARM   MITVRSN.. z/OS 1.4
   +0014  MITUVRSN.           MITUSER..           MITCBV... 01
   +0025  MITENVIR. J         MITLEN... 0050      MITMVRSN. 6
   +0029  MITFLAG1. 28        RSV...... 0000      MITFMID.. HJE7707
   +0034  MITDATE.. 04/25/03  MITTIME.. 08.38     MITMODSZ. 0019D8
   +0044  MITENTAD. 00067860  MITXMAPA. 000678D8  MITAPARN. 000678F8
   Last APAR: OW55429
   Last PTF:  UA01614
$MIT: 0016AC68
   +0000  MITID.... MIT       MITNAME.. HASPARMO  MITVRSN.. z/OS 1.4
   +0014  MITUVRSN.           MITUSER..           MITCBV... 01
   +0025  MITENVIR. V         MITLEN... 0050      MITMVRSN. 6
   +0029  MITFLAG1. A8        RSV...... 0000      MITFMID.. HJE7707
   +0034  MITDATE.. 04/15/02  MITTIME.. 10.39     MITMODSZ. 000288
   +0044  MITENTAD. 0016AE70  MITXMAPA. 0016AEC0  MITAPARN. 0016AEE0
   Last APAR: NONE
   Last PTF:  NONE
$MIT: 00067908
   ...
```

Determine if an address is in a JES2 module

Determine if an address is in a JES2 module



Is an address in a JES2 module?

IPCS **WHERE** command usually doesn't help

- JES2 uses **directed load** into CSA/ECSA so there is no MVS **CDE** (**C**ontents ***D***irectory **E**ntry) control block
  - Most JES2 private modules linked into one gigantic module

Is an address in a JES2 module? *(continued)*

Method #1

- Use what is unofficially called the ***JES2-WHERE*** command
  - **IP CBF addr ASID(X'*jes2-asid*') STR($MODLOC)**

```
00015754: HASPCNVT (X'00015450') + X'00000304'
```

- JES2 ASID not usually required, however
  - Might see **NOT IN ANY JES2 MODULE** if ASID omitted
  - Subsystem might not be called JES2
  - There can be more than one JES2 subsystem (poly-JES)
  - Each JES2 image usually has its own copy of CSA/ECSA modules
- Doesn't work for installation exits using **$ENTRY** with **CSECT=YES**

Is an address in a JES2 module? *(continued)*

Method #2

- Start at an address you want to know about in **IPCS BROWSE** and enter
  F 'MIT ' PREV BDY(8)

Ignore APAR/PTF for *prior* module

MIT

```
00015440   D6E6F5F4   F9F0F940   E4E6F9F4   F5F9F940   | OW54909 UW94599  |
00015450   D4C9E340   C8C1E2D7   C3D5E5E2   A961D6E2   | MIT HASPCNVSz/OS |
00015460   40F14BF2   40404040   40404040   40404040   |  1.2             |
00015470   40404040   01E20050   F6280000   C8D1C5F7   |      .S.&6...HJE7 |
00015480   F7F0F540   F1F061F0   F761F0F2   F1F64BF1   | 705 10/07/0216.1 |
00015490   F0002838   00017BA0   00017C58   00017C78   | 0.....#...@...@. |
```

Module size at +41

Pointer to APAR/PTF number at +4C

- An address is in a JES2 module if
  - it's no more than **module size** beyond the start of the MIT
  - or it's between the start of the MIT and the APAR/PTF #
- Doesn't work for exits using **$ENTRY** with **CSECT=YES**

Is an address in a JES2 module? *(continued)*

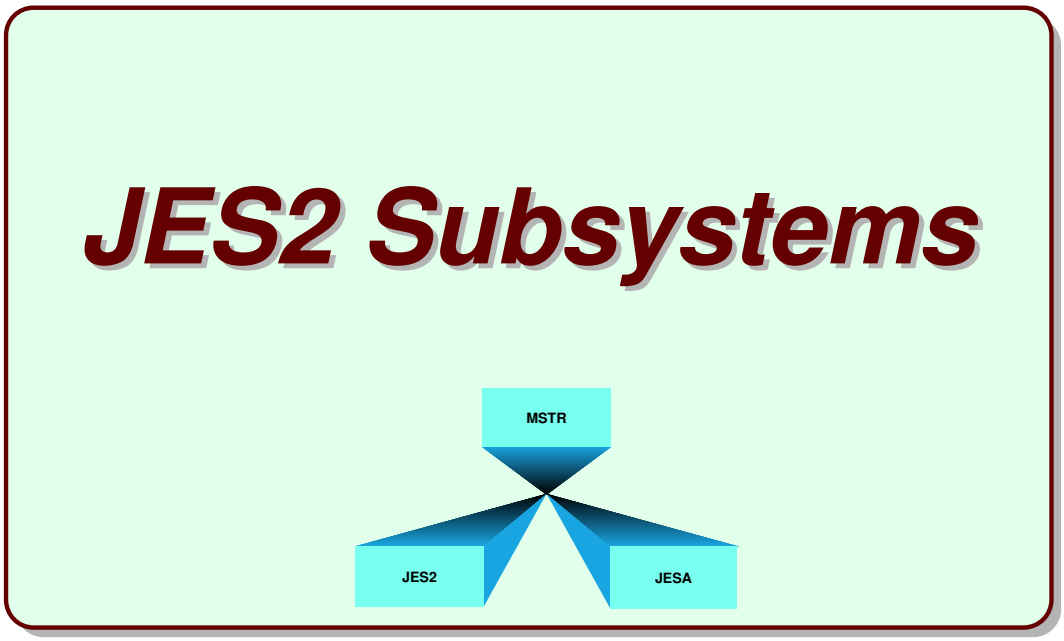
Method #3

- Use if nothing else works
- Go JES2 main IPCS panel and select option 7, then select option for **$LMT**
  - See **Finding JES2 load modules in IPCS** topic
- Look for a **$LMT** for a module containing the address in question, for example

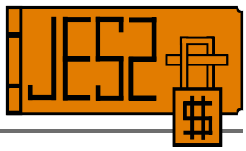
```
$LMT: 37716080
   +0000  LMTMODNM. BAJ2X003  LMTMITAD. 00165B58  LMTSUBPL. 00
   +000D  LMTMODLN. 0004A8    LMTBASEA. 00000000  LMTFLG1.. 28
   +0015  LMTFLG2.. 10        RSV...... 0000      LMTCHAIN. 377160B8
   +001C  RSV...... 00000000
```

Look for a $LMT where address in question is between LMTMITAD value and sum of LMTMITAD + LMTMODLN values

JES2 Subsystems



JES2 subsystems



- JES2 runs as a subsystem under MVS
  - Subsystems defined in **IEFSSNxx** in parmlib
  - ☆ JES2 started under **master subsystem** (**MSTR**) so JES2 proc ***must*** be in master JCL proclib concatenation
  - BCP and application programs request services and information from JES2 using the subsystem interface (SSI)
- Some customers also run **secondary JES2 subsystems**
  - Multiple JES2 images running on one MVS image (also called **poly-JES**)
  - Exploited for both production and test
  - Subsystems may be members of same or different MASes

JES2 subsystems *(continued)*

- Each JES2 subsystem has a command prefix character
  - Defined on CONDEF initialization statement (default **$**)

    CONDEF CONCHAR=$
  - Registered with MVS **CPF** (Command Prefix Facility) service
  - Scope is SYSTEM (default) or SYSPLEX

    CONDEF SCOPE=SYSTEM
  - Example command

    $DMASDEF
- Command prefix character also used as prefix for JES2 messages, for example

  $HASP373 JINKINS STARTED ...

Finding JES2 subsystems

- ***JES2 may not be called JES2***
  - Some customers call the primary JES something else
  - **Primary subsystem** name defined in **IEFSSNxx** parmlib member
  - To display name of primary subsystem in IPCS

IP L CVT+128?+1C L(4)

```
LIST FC99B0. ASID(X'0001') POSITION(X'+1C') LENGTH(X'04') AREA
00FC99CC. D1C5E2F2                          |JES2            |
```

Primary subsystem name

Finding JES2 subsystems *(continued)*

Find all JES2 subsystems in a dump

Run SSCT chain → IP RUNC AD(CVT+128?+18?) LI(4) EX((CBF X MODEL(IEFMSSCT)))

F 'ID. 02' ← 02=JES2 This JES was started at least once since last IPL

```
LIST CDB37C. ASID(X'0001') LENGTH(X'04') AREA

 SSCT: 00CDB37C
    +0000  SSCTID... SSCT      SSCTSCTA. 00CDB310  SSCTSNAM. JES2   <- Subsystem name
    +000C  SSCTFLG1. A0        SSCTSSID. 02        SSCTSSVT. 00CB2130
    +0014  SSCTSUSE. 00CB2D18  SSCTSYN.. 00CDB4C0  SSCTSUS2. 00CB2680

LIST CDB310. ASID(X'0001') LENGTH(X'04') AREA

 SSCT: 00CDB310
    +0000  SSCTID... SSCT      SSCTSCTA. 00CDB334  SSCTSNAM. MSTR
    +000C  SSCTFLG1. 00        SSCTSSID. 00        SSCTSSVT. 00CDB160
    +0014  SSCTSUSE. 00000000  SSCTSYN.. 00CDB52C  SSCTSUS2. 00000000

...

LIST CC7A28. ASID(X'0001') LENGTH(X'04') AREA

 SSCT: 00CC7A28
    +0000  SSCTID... SSCT      SSCTSCTA. 00CC7A4C  SSCTSNAM. JESA   <- Subsystem name
    +000C  SSCTFLG1. 00        SSCTSSID. 00        SSCTSSVT. 00000000
    +0014  SSCTSUSE. 00000000  SSCTSYN.. 00CC7ADC  SSCTSUS2. 00000000

...
```

Assuming this is for a JES, it never completed initialization

Finding JES2 subsystems *(continued)*

Find all JES2 subsystems in a dump (continued)

- More about **SSCTSSID** (set by JES2 and JES3)
  - **00** - Not a JES subsystem or this JES never started (completed initialization)
  - **02** - JES2
  - **03** - JES3
- You can also find information about subsystems using
  - **IP SSIDATA**

List of SSI function codes supported by this subsystem

```
Summary Report for SSIDATA
--------------------------

NUMBER OF DEFINED SUBSYSTEMS =   119
ADDRESS OF SUBSYSTEM REQUEST ROUTER = 00C309A0

SUBSYS = JES2 (PRIMARY)
   DYNAMIC = YES    STATUS = ACTIVE     COMMANDS = NO
   SUBSYSTEM DEFINITION DATA
    SSCVT ADDRESS = 00CDB37C
    USER FIELD 1 = 00CB2D18   USER FIELD 2 = 00CB2680
   SUBSYSTEM VECTOR TABLE DATA
    TOKEN = N/A          ADDRESS = 00CB2130    STATUS = ACTIVE
    FUNC =   1           FUNC =   2          FUNC =   3
...
```

Primary subsystem name is JES2

SSCTSUSE

SSCTSUS2

Finding JES2 subsystems *(continued)*

What subsystem is processing your commands

Who am I talking to?

.DMASDEF ← First find out which member is processing command

```
.HASP843 MASDEF
.HASP843 MASDEF  OWNMEMB=ALTA,AUTOEMEM=ON,CKPTLOCK=ACTION,
.HASP843         COLDTIME=(1996.302,07:34:44),COLDVRSN=OS 1.1.0,
.HASP843         DORMANCY=(50,100),HOLD=50,LOCKOUT=3000,
.HASP843         RESTART=YES,SHARED=CHECK,SYNCTOL=180,
.HASP843         WARMTIME=(2003.012,16:05:36),XCFGRPNM=XYZ,
.HASP843         QREBUILD=0
```

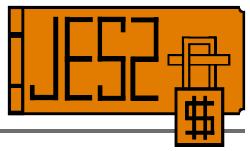
.DMEMBER,STATUS=DEFINED ← Then display that member

```
.HASP673 MEMBER(1
.HASP673 MEMBER(1)    NAME=SYSA,STATUS=ACTIVE,IND=NO,
.HASP673              LASTART=(QUICK,(2003.018,06:58:08)),
.HASP673              TIME=(2003.022,19:38:44.43),
.HASP673              VERSION=z/OS 1.4,SLEVEL=0,SSNAME=JES2,
.HASP673              BOSS=YES
.HASP673 MEMBER(2)
.HASP673 MEMBER(2)    NAME=ALTA,STATUS=ACTIVE,SYSNAME=SYSA,
.HASP673              IND=NO,LASTART=(QUICK,(2003.018,08:18:46)),
.HASP673              TIME=(2003.022,19:38:45.00),
.HASP673              VERSION=z/OS 1.4,SLEVEL=0,SSNAME=JESA,
.HASP673              BOSS=NO
```

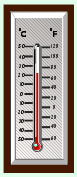
Note; SYSNAME only displayed when MVS name different from JES member name

Commands processed on system SYSA by subsystem JESA

Determining the JES2 Release Level



Determining the JES2 Release Level



Determining JES2 release level *(cont)*

From a dump, method #1

From main JES2 panel, select option for **JES2 base display**

```
*** JES2 Base Display ***

Subsystem "JES2" is in address space ASID(X'018C')
Dump for JES2 release="z/OS 1.4", Product level=34, Service level=0 (pointed
     to by SSCTSUSE), CVTPRODI=HBB7707
Maximum extended region size for "JES2" is 1,746,808K (per LDAELIM)
...
```

JES2 release level (&J2LEVEL)

JES2 product level (&J2PLVL)

JES2 service level (&J2SLVL)

See "Determining the JES2 Release Level" topic in "JES2 Installation Exits" book

Determining JES2 release level *(cont)*

From a dump, method #2

Run SSCT chain

```
IP RUNC AD(CVT+128?+18?) LI(4) EX((CBF X MODEL(IEFMSSCT)))
```

```
LIST AF409C. ASID(X'0001') LENGTH(X'04') AREA
 SSCT: 00AF409C
    +0000  SSCTID... SSCT      SSCTSCTA. 00AF4078  SSCTSNAM. JES2
    +000C  SSCTFLG1. A0        SSCTSSID. 02        SSCTSSVT. 00AD00E0
    +0014  SSCTSUSE. 00AD0D18  SSCTSYN.. 00AE1140  SSCTSUS2. 00AD0630

LIST AF4078. ASID(X'0001') LENGTH(X'04') AREA
 SSCT: 00AF4078
    +0000  SSCTID... SSCT      SSCTSCTA. 00B03040  SSCTSNAM. MSTR
    +000C  SSCTFLG1. 00        SSCTSSID. 00        SSCTSSVT. 00AF2E78
    +0014  SSCTSUSE. 00000000  SSCTSYN.. 00AF4198  SSCTSUS2. 00000000
...
```

Find your subsystem name

Then display data pointed to by SSCTSUSE

```
IP L 00AD0D18 LEN(10)
```

JES2 release level

```
LIST AD0D18. ASID(X'018C') LENGTH(X'0A') AREA
00AD0D18. A961D6E2 40F14BF4 2200             |z/OS 1.4..      |
```

JES2 product level (&J2PLVL)

JES2 service level (&J2SLVL)

Determining JES2 release level *(cont)*

From a dump, method #3

JES2 release level

MIT

```
ASID(X'018C') ADDRESS(7000.) STORAGE ----------------------------------------
Command ===>                                                  SCROLL ===> CSR
00007000   D4C9E340   C8C1E2D7   D5E4C340   A961D6E2   | MIT HASPNUC z/OS |
00007010   40F14BF4   40404040   40404040   40404040   |  1.4             |
00007020   40404040   01E50050   F6280000   C8D1C5F7   |      .V.&6...HJE7 |
00007030   F7F0F740   F0F961F1   F361F0F2   F1F34BF4   | 707 09/13/0213.4 |
00007040   F500AD98   00011840   00011D68   00011D88   | 5..q... .......h |
00007050   E2D740F5   4BF34BF0   40404040   40404040   | SP 5.3.0         |
00007060   F6000BD0   0010D2B0   14B17080   145E60F0   | 6..}..K......;-0 |
00007070   14B17160   00AD05F0   01800000   00009758   | ...-...0......p. |
```

Ignore this constant

- **BROWSE** JES2 address space and locate **MIT** at address **X'7000'**
- JES2 release level is at **MIT+C**

Determining JES2 release level *(cont)*

From **SDSF** use **WHO** command

```
USERID=RMJ,PROC=E52TOOL1,TERMINAL=M05TC403,GRPINDEX=3,GRPNAME=ISFUSER,
MVS=z/OS 01.04.00,JES2=z/OS 1.4,SDSF=HQX7707,ISPF=5.2,RMF/DA=NOTACC,
SERVER=YES,SERVERNAME=SDSF,JESNAME=JES2,MEMBER=AQTS,SYSNAME=AQTS,
SYSPLEX=MCLXCF01,COMM=ENABLED
```

From console

$DMEMBER(SYSA)

```
$HASP673 MEMBER(1)
$HASP673 MEMBER(1)    NAME=SYSA,STATUS=ACTIVE,IND=NO,
$HASP673              LASTART=(QUICK,(2003.205,04:57:41)),
$HASP673              TIME=(2003.208,00:17:00.25),
$HASP673              VERSION=z/OS 1.4,SLEVEL=0,SSNAME=JES2,
$HASP673              BOSS=YES
```

MAS Configurations

MAS Configurations

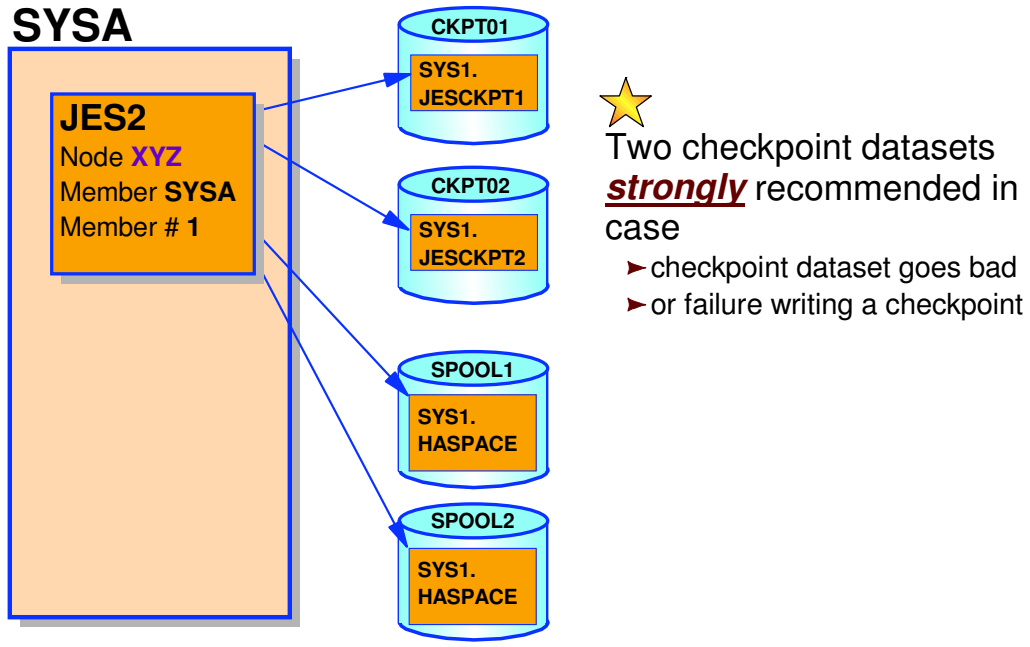
Multi-access SPOOL (MAS)

Multi-Access SPOOL (**MAS**) configuration consists of one or more JES2 subsystems sharing a unique **checkpoint** and **SPOOL**

- A **MAS** may also be called a **JESPLEX**
- ***All MAS members must be in same*** **SYSPLEX**
- ***MAS = Node*** (Node name applies to entire MAS)
- **Checkpoint** dataset contains
  - Job queues (input, output, etc.)
  - MAS-wide resource, configuration, and control data
  - Checkpoint can be on **DASD** or a **Coupling Facility**
- **SPOOL** datasets contain
  - Job input and output
  - Detail job information

☆ ***Checkpoint and SPOOL must stay in sync***

Example single member MAS



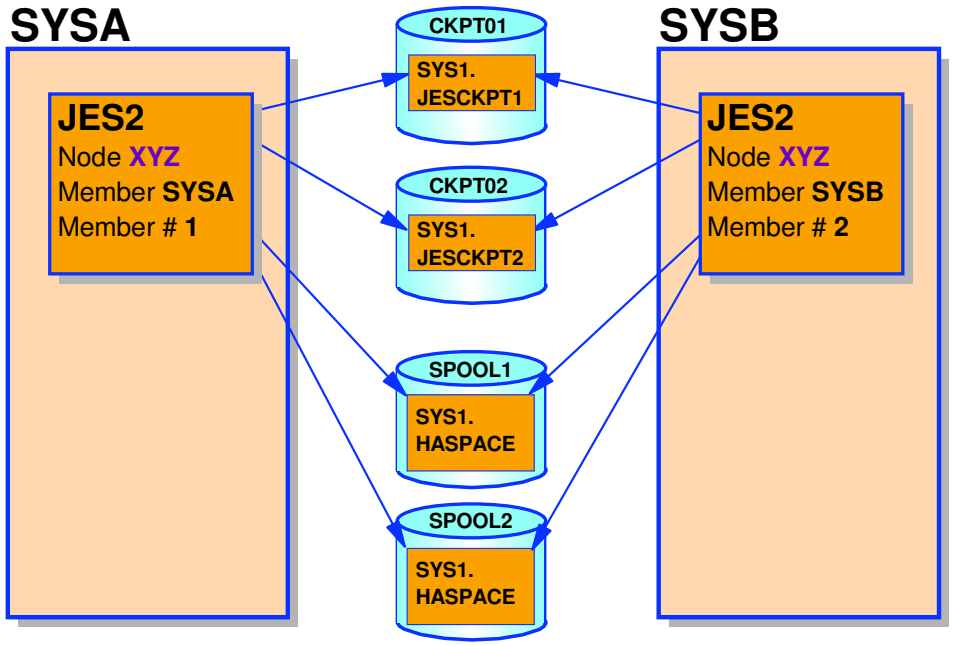
- Two checkpoint datasets ***strongly*** recommended in case
  - checkpoint dataset goes bad
  - or failure writing a checkpoint

Example single member MAS *(continued)*

Related JES2 initialization statements

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

Example two member MAS



Example two member MAS *(continued)*

Related JES2 initialization statements

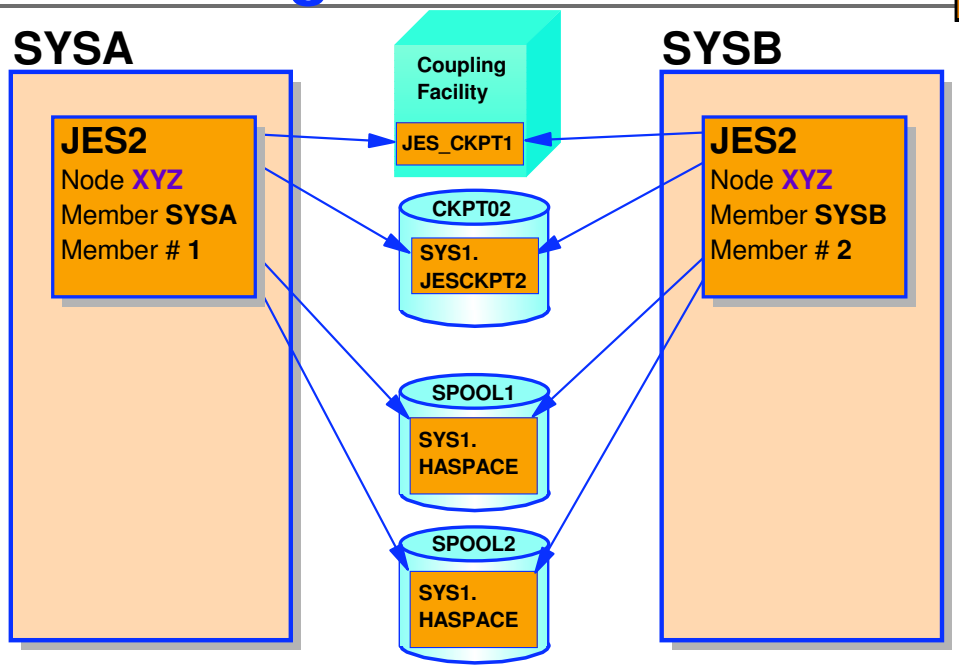
SYSA

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

SYSB

```
MASDEF OWNMEMB=SYSB,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

Example two member MAS using a CF



Example two member MAS using a CF *(cont)*

Related JES2 initialization statements

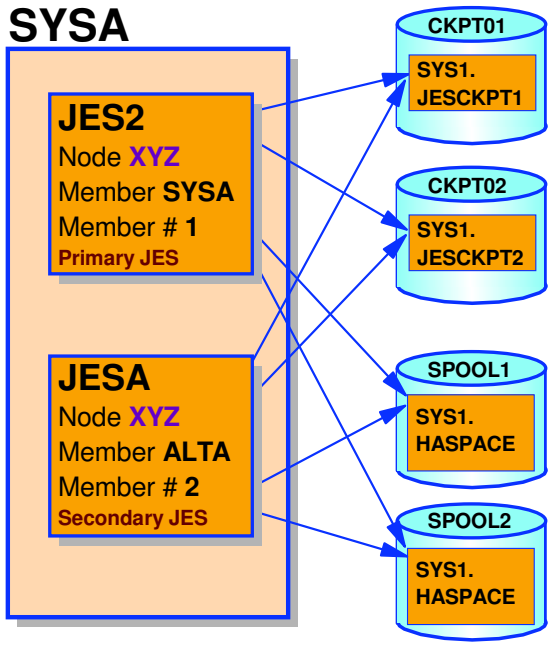
SYSA

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
CKPTDEF CKPT1=(STR=JES_CKPT1),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

SYSB

```
MASDEF OWNMEMB=SYSB,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
CKPTDEF CKPT1=(STR=JES_CKPT1),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

Poly-JES example #1



You can run multiple JES2 subsystems (**poly-JES**) on one MVS image, a **primary JES** and one or more **secondary subsystems**

Poly-JES example #1 *(continued)*

Related JES2 initialization statements

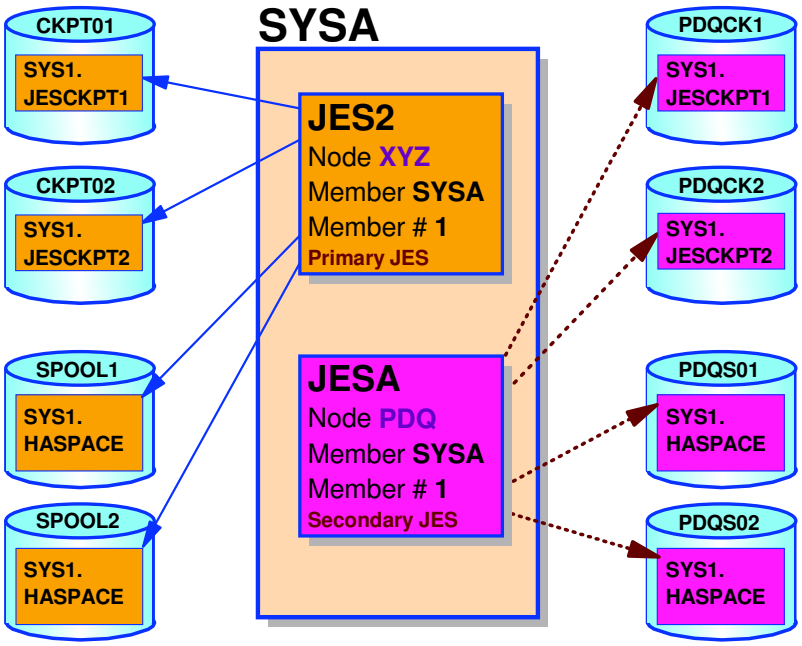
JES2

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=ALTA
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

JESA

```
MASDEF OWNMEMB=ALTA,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=ALTA
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
CONDEF CONCHAR=.,...
```

Poly-JES example #2



Poly-JES example #2 *(continued)*

Related JES2 initialization statements

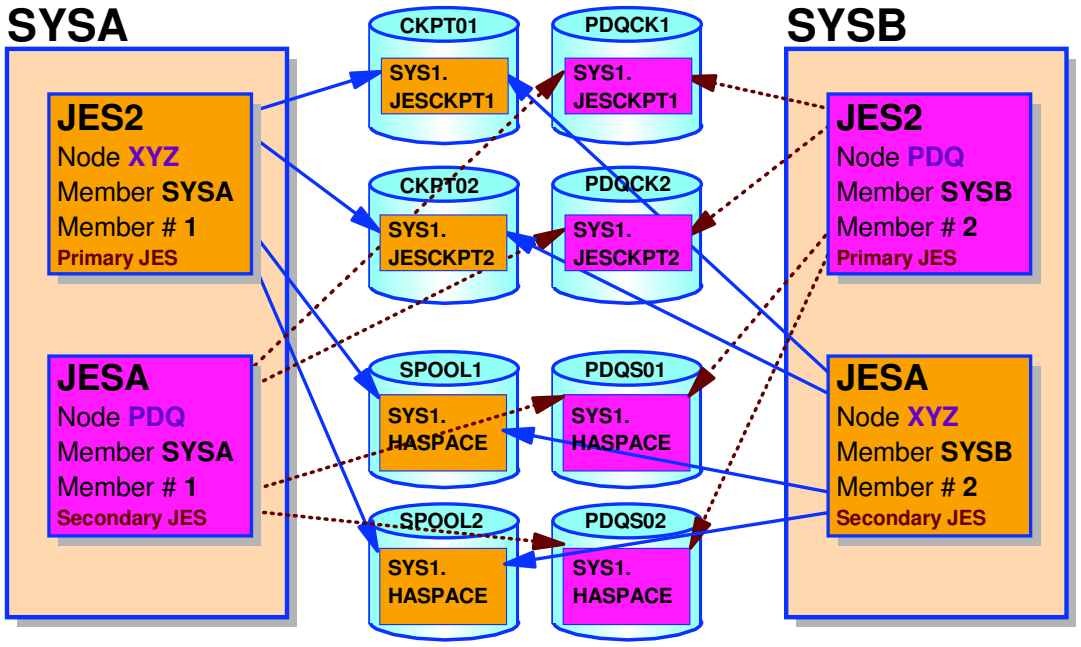
JES2

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
NODE(2) NAME=PDQ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

JESA

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
NJEDEF OWNNODE=2,...
NODE(1) NAME=XYZ,...
NODE(2) NAME=PDQ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=PDQCK1),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=PDQCK2),...
SPOOLDEF V=PDQS,...
CONDEF CONCHAR=.,...
```

Poly-JES example #3



Poly-JES example #3 *(continued)*

Related JES2 initialization statements

SYSA - JES2

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
NODE(2) NAME=PDQ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

SYSA - JESA

```
MASDEF OWNMEMB=SYSA,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=2,...
NODE(1) NAME=XYZ,...
NODE(2) NAME=PDQ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=PDQCK1),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=PDQCK2),...
SPOOLDEF V=PDQS,...
CONDEF CONCHAR=.,...
```

Poly-JES example #3 *(continued)*

Related JES2 initialization statements (cont)

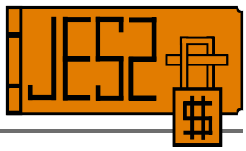
SYSB - JES2

```
MASDEF OWNMEMB=SYSB,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=2,...
NODE(1) NAME=XYZ,...
NODE(2) NAME=PDQ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=PDQCK1),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=PDQCK2),...
SPOOLDEF V=PDQS,...
CONDEF CONCHAR=.,...
```

SYSB - JESA

```
MASDEF OWNMEMB=SYSB,...
MEMBER(1) NAME=SYSA
MEMBER(2) NAME=SYSB
NJEDEF OWNNODE=1,...
NODE(1) NAME=XYZ,...
NODE(2) NAME=PDQ,...
CKPTDEF CKPT1=(DSN=SYS1.JESCKPT1,VOL=CKPT01),
        CKPT2=(DSN=SYS1.JESCKPT2,VOL=CKPT02),...
SPOOLDEF V=SPOOL,...
```

Displaying MAS information



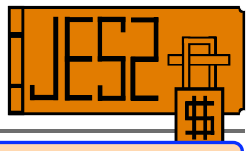
$DMASDEF

```
$HASP843 MASDEF
$HASP843 MASDEF  OWNMEMB=AQTS,AUTOEMEM=ON,CKPTLOCK=ACTION,
$HASP843         COLDTIME=(1996.302,07:34:44),COLDVRSN=OS 1.1.0,
$HASP843         DORMANCY=(50,100),HOLD=50,LOCKOUT=3000,
$HASP843         QUESHELD=AQTS,RESTART=YES,SHARED=CHECK,
$HASP843         SYNCTOL=180,WARMTIME=(2003.012,16:05:36),
$HASP843         XCFGRPNM=PLPSC,QREBUILD=0
```

Selected MAS-wide attributes

Keyword	Description
OWNMEMB	Member creating this display
COLDSTART	Date/time (GMT) of last JES2 cold start
COLDVRSN	JES2 release that performed the cold start
WARMTIME	Date/time (GMT) of last All-Member Warm Start
XCFGRPNM	XCF group name for this MAS

Displaying MAS information *(continued)*



$DMEMBER,STATUS=DEFINED

```
$HASP673 MEMBER(1)
$HASP673 MEMBER(1)    NAME=AQFT,STATUS=ACTIVE,IND=NO,
$HASP673              LASTART=(QUICK,(2003.032,06:42:49)),
$HASP673              TIME=(2003.038,22:17:59.98),
$HASP673              VERSION=z/OS 1.4,SLEVEL=0,SSNAME=JES2,
$HASP673              BOSS=YES
$HASP673 MEMBER(2)
$HASP673 MEMBER(2)    NAME=AQTS,STATUS=ACTIVE,IND=NO,
$HASP673              LASTART=(QUICK,(2003.037,05:59:50)),
$HASP673              TIME=(2003.038,22:17:59.41),
$HASP673              VERSION=z/OS 1.4,SLEVEL=0,SSNAME=JES2,
$HASP673              BOSS=YES
```

Selected member attributes

Keyword	Description
NAME	JES2 member name
STATUS	Member status
SYSNAME	MVS system name (**only displayed if different from JES name**)
LASTART	Last JES2 start type and time (GMT)
TIME	GMT time of last checkpoint update (known to displaying member)
VERSION	JES2 release on member
SSNAME	JES2 subsystem name

Displaying MAS information *(continued)*

$DCKPTDEF

```
$HASP829 CKPTDEF
$HASP829 CKPTDEF  CKPT1=(DSNAME=SYS1.HASPCKPT,VOLSER=CHKPT0,
$HASP829          INUSE=YES,VOLATILE=NO),
$HASP829          CKPT2=(DSNAME=SYS1.HASPCKPT,VOLSER=CHKPT3,
$HASP829          INUSE=YES,VOLATILE=NO),
$HASP829          NEWCKPT1=(DSNAME=SYS1.BKUPCKPT,VOLSER=SYSDA1),
$HASP829          NEWCKPT2=(DSNAME=SYS1.BKUPCKPT,VOLSER=SYSAQ1),
$HASP829          MODE=DUPLEX,DUPLEX=ON,LOGSIZE=5,
$HASP829          VERSIONS=(STATUS=ACTIVE,NUMBER=10,WARN=80,
$HASP829          MAXFAIL=0,NUMFAIL=0,VERSFREE=10,MAXUSED=2),
$HASP829          RECONFIG=NO,VOLATILE=(ONECKPT=WTOR,
$HASP829          ALLCKPT=WTOR),OPVERIFY=YES
```

Selected checkpoint attributes

Keyword	Description
CKPTn	Checkpoint dataset locations and status
NEWCKPTn	Replacement checkpoint datasets for use in reconfigurations
MODE	DUAL (alternating dataset usage) or DUPLEX (primary/secondary)
DUPLEX	Whether this member writes CKPT2 when in MODE=DUPLEX
OPVERIFY	Whether operator concurrence required in reconfigurations for I/O errors

Displaying MAS information *(continued)*

$DSPOOLDEF

```
$HASP844 SPOOLDEF
$HASP844 SPOOLDEF  BUFSIZE=3992,DSNAME=SYS1.HASPACE,
$HASP844           FENCE=(ACTIVE=YES,VOLUMES=1),GCRATE=NORMAL,
$HASP844           LASTSVAL=(2001.161,14:35:06),RELADDR=NEVER,
$HASP844           SPOOLNUM=32,TGSIZE=30,TGSPACE=(MAX=276896,
$HASP844           DEFINED=231000,ACTIVE=214500,PERCENT=73.5263,
$HASP844           FREE=56786,WARN=85),TRKCELL=6,VOLUME=SPOL1
```

Selected SPOOL attributes

Keyword	Description
DSNAME	DSNAME of SPOOL datasets
FENCE	Used to (attempt) to limit job's SPOOL space to subset of volumes
LASTSVAL	Date/time (GMT) of last All-Member Warm Start with SPOOL=VALIDATE
SPOOLNUM	Current maximum number of SPOOL volumes
TGSPACE	Track Group info (space assign to jobs in units of Track Groups)
VOLUME	SPOOL volser prefix (SPOOL volsers must begin with this prefix)

Displaying MAS information *(continued)*

$DNJEDEF

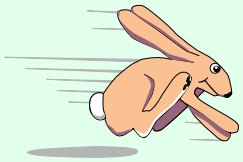
```
$HASP831 NJEDEF
$HASP831 NJEDEF  OWNNAME=PLPSC,OWNNODE=1,DELAY=180,
$HASP831         HDRBUF=(LIMIT=144,WARN=80,FREE=143),JRNUM=3,
$HASP831         JTNUM=3,SRNUM=4,STNUM=4,LINENUM=60,MAILMSG=NO,
$HASP831         MAXHOP=0,NODENUM=6000,PATH=1,RESTMAX=262136000,
$HASP831         RESTNODE=100,RESTTOL=0,TIMETOL=1440
```

Selected NODE attributes

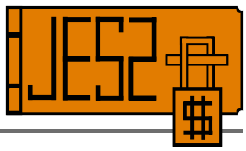
Keyword	Description
OWNNAME	JES2 node name for this MAS
OWNNODE	JES2 node number for this MAS

JES2 Start Types

JES2 Start Types



JES2 start types



- What the operator can request
  - **COLD**
  - **COLD,FORMAT** or just **FORMAT** which implies **COLD**
  - **WARM**
- Resulting JES2 start types
  - ***Cold*** (with or without a FORMAT)
  - ***All-Member Warm Start***
  - ***Single-Member Warm Start***
  - ***Quick Start***
  - ***Hot Start***

JES2 start types *(continued)*

- Start type reported by initialization messages
  - **$HASP493 *subsys start-type* IS IN PROGRESS {*text*}**
  - **$HASP492 *subsys start-type* HAS COMPLETED {*text*}**
- A member can also be **restarted** by another member as result of
  - **$EMEMBER** command (must be MVS **system-gone**)
  - Automatic restart specifications on **MASDEF** initialization statement

When one member is "restarted" by another, we call this a "surrogate restart" in JES2 jargon. The work the failed member was processing is cleaned up and unbusied to make it available for selection and processing by other MAS members (assuming they are otherwise eligible to process the work)

Start type descriptions

- ***Cold Start***
  - ***Any existing jobs on checkpoint and SPOOL vanish***
  - Checkpoint dataset(s) are formatted
  - SPOOL may be formatted
    - If **FORMAT** specified
    - or if sample reads at start and end of any SPOOL extent indicate formatting needed
  - JES2 must not be hot-startable on any member
    - JES2 must have come down clean or the MVS it was on must have left the SYSPLEX (MVS **system-gone**)
    - Practically speaking, to perform a cold start, these restrictions usually require **SYSPLEX IPL** and specifying **COLD** on the first JES2 to start
  - ☆ Make sure JES2 init parms reflect changes to MAS wide values made using $T commands since last cold start

Start type descriptions *(continued)*

- **All-Member Warm Start**
  - Performed by first member to start after all members either
    - ended cleanly with $PJES2
    - or their systems left the SYSPLEX (MVS **system-gone**)
  - Another start type applies if JES2 is up or hot-startable on any member (including this member)
  - ☆ Isn't all-member warm start unless **HASP493** says so
  - All-member warm start required for some checkpoint reconfiguration actions

Start type descriptions *(continued)*

- **All-Member Warm Start *(continued)***
  - Performs some verification and cleanup not done on any other start type
  - ☆ May take a while because JES2 reads **IOT** chains for all jobs
    - Over 500,000 I/Os for some large customers
    - ☆ Number of I/Os depends on applications ... and includes IOTs for null SYSOUT datasets
    - I/O done in parallel by multiple **Warm Start PCEs**. Number of PCEs determined by algorithm based on number of selectable SPOOL volumes
  - **SPOOL=VALIDATE** start option only honored on **all-member warm start**
    - Little reason to use in today's world
    - **$TSPOOLDEF,GCRATE=FAST** usually better alternative (new in **OW53863**)

Start type descriptions *(continued)*

- **Single-Member Warm Start**
  - Performed on JES2 start after IPL for member which didn't shut down cleanly
  - JES2 cleans up and unbusies work member was processing when it went down

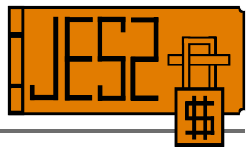
Start type descriptions *(continued)*

- **Quick Start**
  - Fast JES2 start
    - on first IPL of member after another member performed an **all-member warm start**
    - on JES2 start after this member shut down cleanly and at least one other member is either running or is hot-startable
    - on JES2 start after IPL following the **surrogate restart** of this member by another as the result of
      - **$EMEMBER** command
      - or **MASDEF AUTOEMEM=ON** specification for this member and **MASDEF RESTART=YES** on member performing the surrogate restart
    - on first start of a new member
  - **Quick start** not done when **all-member warm start** can be done instead

Start type descriptions *(continued)*

- **Hot Start**
  - Restart of JES2 after a failure without an intervening IPL
  - Failure may have been unexpected, or the result of a **$PJES2,ABEND** or **$PJES2,ABEND,FORCE**
  - Executing jobs continue to run while JES2 is down until they request JES2 services, at which point they wait
  - Any executing jobs waiting on JES2 services resume when JES2 is restarted
  - Jobs actively being submitted through internal readers fail
  - JES mode (non-FSS) printers, NJE, and RJE devices must be restarted ... all other activity either continues or waits for JES2 to restart

JES2 address space and associates



JES2 address space and associates

JES2 address space and associates

Address spaces

JESXCF

1 per MVS

jes2

1 per JES2 subsystem

jes2AUX

1 per JES2 subsystem

jes2MON

1 per JES2 subsystem

New in z/OS 1.4

Associated data spaces

JES00001
JES00002
JESnnnnn

n = 1 + 1 per JES subsystem

00001IEF
00002IEF
nnnnnIEF

n = 1 per JCL Converter

JES2IRDS
JES2CKVR
JES2PSO
JES2SAPI
JES2STAC
JES2TJEV

New in z/OS 1.4

Owned / used by MVS Converter component

JES2 address space and associates *(cont)*

- JESXCF address space
  - Server address space, one per MVS image
  - BCP component used by both JES2 and JES3
  - Started by MVS at IPL
  - If JESXCF terminates, JES2 terminates
    - Next JES2 restart causes JESXCF restart
    - JES3 cannot restart JESXCF, IPL required
  - Functional layer above XCF, requires SYSPLEX
  - Provides communications between members
- JESXCF dataspaces
  - One dataspace for JESXCF usage
  - Additional dataspace for each JES subsystem on this member (**JES00002** - **JES*nnnnn***)

JES2 address space and associates *(cont)*

- JES2 address space
  - One per JES2 subsystem
  - Same name as subsystem, usually **JES2**
  - Manages most aspects of JES2 operation
  - Can ABEND and restart without executing jobs failing
- JES2 dataspaces
  - One dataspace per JCL Converter subtask
  - Up to 10 dataspaces (***nnnnn*IEF**)
  - Owned and used by **MVS Converter component**
  - Used to buffer JCL

JES2 address space and associates *(cont)*

- JES2AUX
  - One per JES2 subsystem
  - Called ***jes2*AUX** where ***jes2*** is JES2 subsystem name, usually **JES2**
  - Anchor for system **LX** and dataspaces
  - In own address space because cross-memory functions and dataspaces ***must*** survive JES2 hot starts

JES2 address space and associates *(cont)*

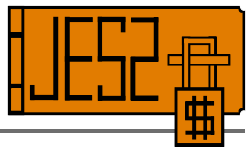
- JES2AUX dataspaces
  - Names begin with subsystem name

DSPNAME	Description
***jes2*IRDS**	Internal reader buffers (2 x 8K per INTRDR)
***jes2*CKVR**	Checkpoint versions (up to 50)
***jes2*PSO**	**$PSOs** ($CPOOL managed) ***New*** in z/OS 1.4
***jes2*SAPI**	**$SAPIDs** ($CPOOL managed)
***jes2*STAC**	**$STACs** ($CPOOL managed) ***New*** in z/OS 1.4
***jes2*TJEV**	**TJEVs** ($CPOOL managed)

JES2 address space and associates *(cont)*

- JES2MON
  - **New** in **z/OS 1.4**
  - One per JES2 subsystem
  - Called ***jes2MON*** where ***jes2*** is JES2 subsystem name, usually **JES2**
  - Starts when JES2 first initializes, ends on clean JES2 shutdown ($PJES2)
  - Automatically restarted within few minutes if monitor stopped or fails
  - **HASJ2MON** module loaded into JES2 during initialization
    - Copied to JES2MON address space when monitor (re)starts
    - Monitor restarted if new version loaded on hot start

JES2 program properties



JES2 program properties

JES2 program properties

IBM supplied JES2 **PPT** attributes ***shipped*** in BCP module **IEFSDPPT**

- **HASJES20** attributes ***expressed*** in **SCHEDxx** parmlib notation

PPT PGMNAME(HASJES20) NOCANCEL NOSWAP NODSI PASS SYST KEY(1)

NOCANCEL	Non-cancelable
NOSWAP	Non-swappable
NODSI	No dataset integrity
PASS	Cannot bypass security
SYST	System task
KEY(1)	Use PSW key 1

JES2 program properties *(continued)*

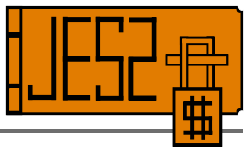
Finding **PPT** entries in a dump

- In BROWSE issue
  L CVT+128?+68?
- And look for **HASJES20**

```
38E5A210   D7D7E340   01000020   001000C4   00DD0000   | PPT .......D.... |
...
38E5A2E0   C8C1E2D1   C5E2F2F0   EC10FFFF   00800000   | HASJES20........ |
```

- **CVTJESCT** (+128) => **JESPPT** (+68) => **PPT**
- Mapped by PPT data area (IEFZB610)
- **JES2 Base Display** in IPCS reports error if JES2 swappable
  - **IP L *ascb_addr*+90?+11 ASID(X'*jes2_asid*')**
  - Error if X'80' bit off in OUCBSFL in OUCB

JES2 structure

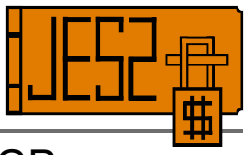


JES2 structure

JES2 private area

- **HASJES20** load module
  - Contains **HASPxxxx** modules
  - JES2 entry point, main task, and subtasks
  - Loaded at address X'7000'
  - Begins with module **HASPNUC**
    - $MIT for HASPNUC (at 7000) is also start of $HCT
  - **HASPINIT** load module
    - Loaded by HASJES20
    - Contains **HASPIRxx** modules
    - Deleted at end of initialization
- **$HCT** (**H**ASP **C**ontrol **T**able)
  - Primary data area for address space
  - Pointed to by **$HCCT** in common

JES2 main task



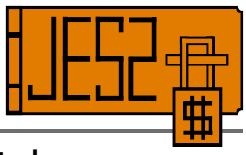
Most JES2 work done under main task TCB

- Main task problems affect entire system
- Entire MAS affected if checkpoint owned by this member
- JES2 health monitor (new in z/OS 1.4) currently monitors only the main task
- 4th TCB in JES2 address space
- Find **HASJES20** in IPCS display for
  - **IP SUMM FORMAT JOBN(JES2)**
- TCB pointed to by **$HASPTCB** (in $HCT) and **CCTHTCBA** (in $HCCT)

JES2 main task *(continued)*

- Main task should ***not*** MVS WAIT except in JES2 dispatcher
  - Branch entry wait at single point in HASPNUC
  - PSW points to label **HASPWAKE** when WAITing
    - Usually the PSW address when JES2 dumped by command or for asynchronous ABENDs and SRB-to-TCB-percolation
  - WAITs at other points can cause SYSPLEX hangs
- Functions that might WAIT are given to subtasks
  - Such as RACROUTE, DYNALLOC, SMFEWTM, and WTO
  - Subtask queuing implicit for services such as **$WTO**
  - Exceptions include WTOs for critical main task functions like checkpoint (HASP263)

JES2 main task *(continued)*



Main task sub-dispatched by JES2 dispatcher

- Processes managed using **$PCE**s (**P**rocessor **C**ontrol **E**lements)
  - One PCE for each function or device (CKPT, L41.ST1)
  - Some functions have multiple PCEs (CNVT)
  - Number of PCEs typically range from 100 to 50,000
- JES2 dispatcher is synchronous
  - Branch entry
  - No MVS hooks or LPSW
- PCEs don't MVS **WAIT**, they **$WAIT** (**BASR** to JES2 dispatcher)
- Serialization implicit between $WAITs (non-JES2-preemptable)
- Recovery also sub-dispatched ($ESTAE, $SETRP)

JES2 main task *(continued)*

Main task WAIT / POST mechanisms

- **$WAIT** / **$POST** used in main task (not real WAIT/POST)
- **$$POST** used by subtasks and cross memory (real POST)
- Main task waits on single ECB pointed to by
  - **CCTPOSTE** (1st word, **CCTHECBA**) in $HCCT
  - **$HASPECB** in $HCT
- Main task, however, waits for hundreds of events
  - JES2 exploits MVS extended ECBs using **$XECBs**
  - POST exit in HASPNUC posts main task ECB

When JES2 address space not in dump

- Use JES2 base display for exceptions
- Look for **STCK** time stamps in base display
  - **CCTJ2WAT** is last time main task entered its normal wait at label **HASPWAKE**
  - **CCTJ2DSP** is last time JES2 main task woke up from its normal wait
- Get main task TCB address from field **CCTHTCBA** and look at main task in system trace using

  IP SYSTRACE ASID(X'j*es2_asid*) TCB(X'*tcb_addr*') TIME(GMT)

JES2 main task in traces

- Look at JES2 dispatcher **CTRACE** using
  IP CTRACE COMP(SYS*jes2*) SUB((DISP)) FULL
  - Where ***jes2*** is JES2 subsystem name
  - Contains entries created by JES2 dispatcher
- Look at main task in **system trace** using
  IP SYSTRACE ASID(X'*jes2_asid*) TCB(X'*tcb_addr*') TIME(GMT)
  - If system trace large enough you can see recent PCE dispatches and $WAITs. Look for **PC** entries for MVS **TIMEUSED service**

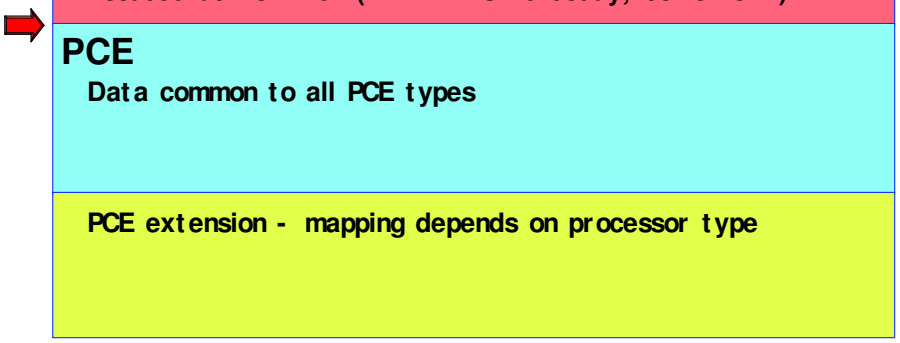
PC number for TIMEUSED

```
00-002D 008E9E88   PC     ...   1      00009228            00316
```

Address where PC was invoked. Use HASPNUC assembly to distinguish PCE dispatches from $WAITs

$PCE data area

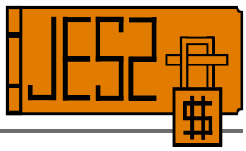
- PCE (**P**rocessor **C**ontrol **E**lement)
- Pointed to by **R13** and **$CURPCE** (in $HCT) when dispatched



$PCE data area *(continued)*

- Prefixed by **$PPB** (PPB DSECT in **$PERFCB**)
- Data common to all PCE types
  - If PCE address not in **$CURPCE**, then PCE is $WAITing
  - Begins with 18 word O/S style save area with eyecatcher of PCE
    - Contains registers when PCE is $WAITing, otherwise ***don't*** use because values are very processing dependent
    - **R15** value at +10 is the resume address when $WAITing
  - **PCEID** identifies PCE type
    - ☆ You can find names of mapping macros for PCE extensions at end of $PCE macro
  - **PCEWTTIM** contains STCK (GMT) time when PCE last $WAITed
  - **PCEJQE** may contain a **$JQE** address

$PCE displays in IPCS



- Using CBF command

  IP CBF addr STR($PCE)

  - Displays PCE without related data
- **JES2 base display** formats current PCE and related data
  - Based on non-zero **$CURPCE** in $HCT)
- Using "**JES2 processors**" panel invoked from option **5** on main JES2 panel
  - Optionally formats related data

Example $PCE display in IPCS

Eyecatcher

Only use registers when PCE is $WAITing (PCE address not in $CURPCE), otherwise occasionally useful in specific code paths

```
$PCE: 16C7A528
   +0000  PCEEYE... PCE
   +0000  PSVID.... PCE       PSVPREV.. 00000000  PSVNEXT.. 16C00AF8
   +000C  PSVR14... 800A7724  PSVR15... 000AFD88  PSVR0.... 268901A0
   +0018  PSVR1.... 00000000  PSVR2.... 00000000  PSVR3.... 00000000
   +0024  PSVR4.... 00059FF8  PSVR5.... 16C1A124  PSVR6.... 00331B06
   +0030  PSVR7.... 2689019F  PSVR8.... 17B491A4  PSVR9.... 00000000
   +003C  PSVR10... 17163008  PSVR11... 00007000  PSVR12... 000A7220
   +0048  PSVADDR.. 16C7A528  PSVLABAD. 16C00AF8  PSVLSPTR. 00000000
   +0054  PSVARPTR. 00000000  PSVMODE.. 01        PSVEXID.. 00
   +005A  RSV...... 00000000  0000
   +0060  PSVSTCK.. 00000000  00000000
   09/17/2042 23:53:47.370496
   +0048  PCEDOM68. 16C7A528
   +004C  PCELPSV.. 16C00AF8  PCEXITID. 00
   ...
```

Exit number if running in an exit or a service invoked by an exit

Example $PCE display in IPCS *(continued)*


```
...
+0068  PCEPREV.. 16C7CA90  PCENEXT.. 16C7AA90  PCEPCEA.. 00007B58
+0074  PCEPCEB.. 00007B58  PCEERA... 00000000  PCEPRE... 26890168
+0080  PCEEWF... 80        PCEFLAGS. 00        PCEFLAG2. 22
+0083  PCEFLGCS. 00        PCENDSPC. 0000      ⇨PCEID.... 8107
+0088 ⇨PCEUSER0. 00000000 ⇨PCEUSER1. 00000000  RSV..... 00000000
+0098  PCEWTTIM. B9D14CA8  B286FC41
08/03/2003 22:09:26.841455
+00A0  PCEDADCT.           PCEDCTPC. 16C7A528  PCEDCTFL. 00000000
+00A8  PCEBUFAD. 00000001  PCESEEK.. 00331B06  PCEIOEWF. 16C7A528
+00B4  PCEBUFCN. 0000      PCEDEVTP. 00        RSV..... 00
+00B8 ⇨PCELENG.. 0528      PCEROLOQ. F2        PCESEQ... 08
+00BC ⇨PCEDCT... 171712B8 ⇨PCEJQE... 17B491A4  PCEJOBID. ........
+00C8  PCEJQEIX. 00000000  PCEPTAB.. 0008A650  PCEFSACB. 00000000
+00D4  PCEWAVE.. 16C83718  PCENTITY. 16C83A68  PCECONCT. 00000000
+00E0 ⇨PCEACTCT. 00000001  RSV..... 00000000  PCEWORKA. 16C7A618
+00EC  RSV..... 00000000
      ***** PRINT/PUNCH PCE WORK AREA *****
+00F0  PPPFLAG1. 04        PPPFLAG2. 10        PPPFLAG3. 01
...
```

Last $WAIT time (GMT)

PCE Performance Block


```
$PPB: 16C7A4E8
   +0000  PPBID.... PPB       PPBVERN.. 01        RSV..... 00
   +0006  PPBLPOST. 0120      PPBPTPB.. 16C28260  PPBWAITC. 00001277
   +0010  PPBCPUT.. 00000000  00064B84
   +0018  PPBRUNT.. 00000000  0006CF7B
   +0020  PPBWAITT. 0000000D  274A5ACB
   +0028  PPBQSUSE. 00000000  0001812C            PPBIOCNT. 00001090
   +0034  PPBCKPTN. 000004D1  RSV..... 00000000  00000000
```

Located at **PCE - 40**

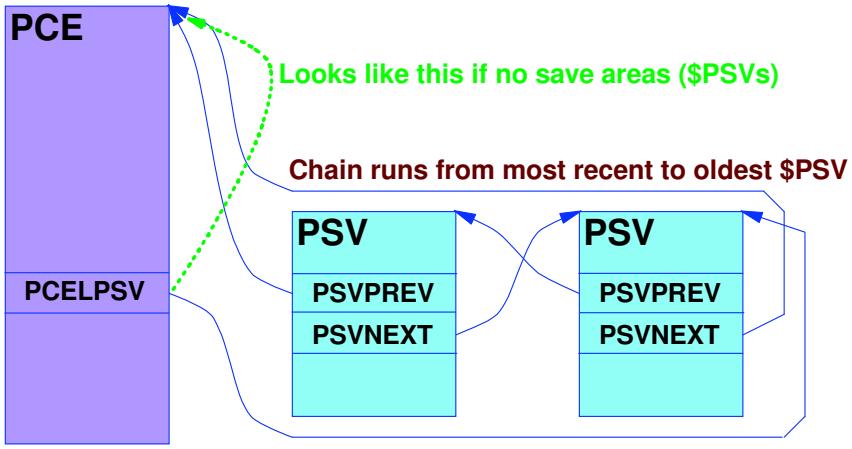
Contains JES2 **PERFDATA** (see **$PERFCB** data area)

- Pointer to other PERFDATA info
- CPU time in microseconds
- Run time in microseconds
- $WAIT time in microseconds
- I/O count, etc.

Main task save areas

$PSV (**P**rocessor **S**a**v**e Area)

- Obtained / freed by **$SAVE** / **$RETURN** macros
- Chained from owning $PCE



Main task save area display in IPCS

$PSVs may be formatted in

- **JES2 base display** following current PCE
- Using "**JES2 processors**" panel invoked from option **5** on main JES2 panel (as optional PCE related data)

Main task save area display in IPCS *(cont)*


```
$PSV: 16C00AF8
   +0000  PSVID.... SAVE      PSVPREV.. 16C7A528  PSVNEXT.. 16C7A528
   +000C  PSVR14... 800B1A1C  PSVR15... 00331B06  PSVR0.... 17B491A4
   +0018  PSVR1.... 001D1000  PSVR2.... 00000001  PSVR3.... 001D1000
   +0024  PSVR4.... 260AF000  PSVR5.... 00331B06  PSVR6.... 00000000
   +0030  PSVR7.... 800B191A  PSVR8.... 00000000  PSVR9.... 00000000
   +003C  PSVR10... 001D0000  PSVR11... 00007000  PSVR12... 000BD02C
   +0048  PSVADDR.. 16C7A528  PSVLABAD. 00156400  PSVLSPTR. 7F7BB010
   +0054  PSVARPTR. 00000000  PSVMODE.. 01        PSVEXID.. 00
   +005A  RSV...... 00000000  0000
   +0060  PSVSTCK.. B9D14CA8  B2B61F81
   08/03/2003 22:09:26.842209
   Routine name: $DISTERR
      00156400: HASPRAS  (X'00156228') + X'000001D8'
   Address routine called from (assuming normal linkage):
      000B1A1C: HASPPRPU (X'000AFF98') + X'00001A84'
*** ERROR:  $DISTERR invoked from symbol PHASP185 in CSECT HASPPRPU
1 $PSV(s) processed
```

Exit #

New in z/OS 1.4
STCK time at $SAVE

Assumption not always good

JES2 processor panel (Option 5)


```
------------------------- JES2 Processor List --------------------------------
OPTION ===>                                              SCROLL ===> CSR
Enter JES2 name ===> JES2
  0 Select related control blocks to include in display
Select processor(s) by activity or status:
  A All PCE types                    E Ended PCEs
  B PCE at address --> 00000000      F PCEs $WAITing at address --> 00000000
  C Current PCE                      G Ready PCEs
  D Active PCEs ($ACTIVE)            H Ready PCEs in reverse order
                                                          More:     +
Select processor(s) by type:
  1 Input
  2 Internal Reader
  3 Remote Reader
  4 Asynchronous
  5 JCL Conversion
  6 Execution
  7 Process SYSOUT
  8 Output
  9 Local Printer
 10 Remote Printer

Enter UP and DOWN commands to scroll the list of options.
Enter END command to terminate.
```

Tip: If one PCE is overlayed, others may be also. Since this report is usually very small, PPB and PCE eyecatcher overlays are easy to spot

Processor related control block selection


```
-------------- JES2 Processor Related Control Block List --------------------
OPTION ===>

Specify related control blocks to include in $PCE displays:

   $WAIT   --> Y  (Y or N)
   $PSV(s) --> Y  (Y or N)
   $DCT    --> Y  (Y or N)
   $JQE    --> Y  (Y or N)









Enter END command to terminate.
```

PCE example showing $WAIT

$WAIT information shown in two formats

- Module and offset
- Module and sequence number in source (colums 73 - 80)

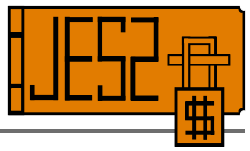
```
*** Checkpoint $PCE(s) ***

$PCE number 1:
 $PCE: 16C6EDC8
    +0000  PCEEYE... PCE
    +0000  PSVID.... PCE       PSVPREV.. 00000000  PSVNEXT.. 16C6EDC8
    ...
    +0218  CKPECMBF. B9D14CA5  CKPECNID. 9D908D80  CKPECART. 00000000
    +0224            00000000  CKPKITPS. 00000000
    PCE is currently $WAITing at:
       0009AAAE: HASPCKPT (X'00099A68') + X'00001046'
$WAIT: 0009AAAE
    +0004  $WTFLAG1. 90        $WTINHBT. 80
    +0006  $WTCSECT. HASPCKPT  $WTSEQF.. 12260000  $WTRESQO. FFF0
0 $PSV(s) processed

1 $PCE(s) processed
```

Sequence field in source (not an address)

Where to find PCE info

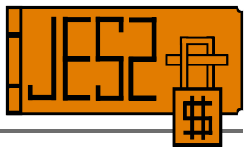
- **$PCE** macro
  - Detailed macro prolog
  - List of **PCEIDs** and names of extensions (**$xxxWORK**)
- Short/long names of all PCEs, entry point names, and other attributes
  - Look at PCE table in module **HASPTABS** between
    - $PCETAB TABLE=HASP
    - $PCETAB TABLE=END
  - Entry points indirectly identified using symbols in **$MODMAP** macro

Job Queue Elements & other checkpoint data



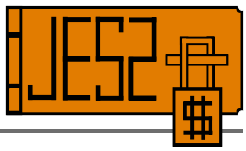
Job Queue Elements and other checkpoint data

Job Queue Elements



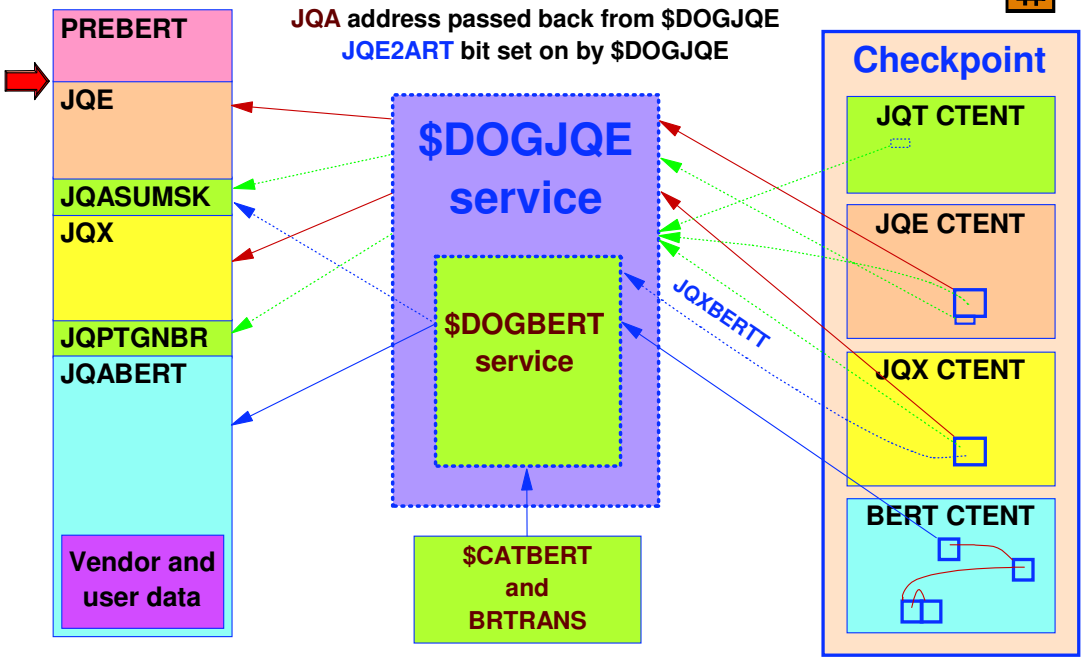
- **$JQEs** reside in checkpoint
- Primary control block representing a job
- Moves from queue to queue as job moves from job phase to phase (input, conversion, execution, etc.)
- Queue heads reside in checkpoint
  - Heads for all queue types in **checkpointed $HCT**
  - For jobs in execution or awaiting the execution, ***real*** queue heads are in $CATs
    - **$CATs** (**C**lass **A**ttribute **T**able) in checkpoint in **BERTs**
    - IPCS uses checkpointed HCT queue heads only

Job Queue Elements

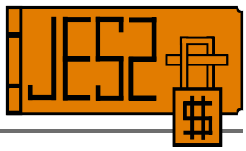


- JQE is actually a collection of data
  - **"Real" JQE** (in JQE section of checkpoint)
  - **JQX** (JQE extension in JQX section of checkpoint)
  - **BERT** resident data (in BERT section of checkpoint)
  - JQE extension for track group counts (also in ckpt)
- Corresponding symbol prefixes in **$JQE** data area
  - **JQExxxxx**
  - **JQXxxxxx**
  - **JQAxxxxx**
- JQE almost always viewed as a composite called a **JQA** or **Artificial JQE**
  - Built by **$DOGJQE** service (uses **$DOGBERT** service)

Artificial JQE (also called JQA)

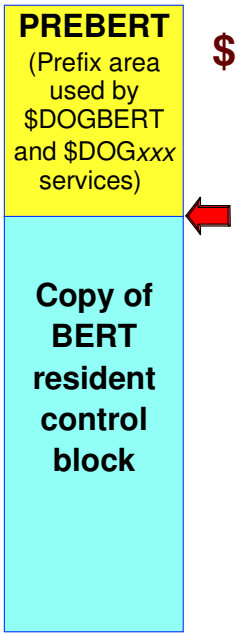


What are BERTs?



- **B**lock **E**xtent **R**euse **T**able
- Used to extend checkpointed control blocks without a cold start
- General-purpose X'40' byte checkpoint elements
- Number control by **CKPTSPACE BERTNUM**
- BERTs currently exploited for
  - **JQA** (extension of $JQE)
  - **$CAT**
  - **$WSC**
  - Vendor and user extensions, but havn't seen new CBs
- Data accessed by **$DOGxxx** services and defined by **$BERTTAB** tables

$DOGBERT



$DOGBERT service encapsulates access

- Moves data to and from BERTs in checkpoint
  - Callers work with a copy of the data
  - Data in checkpoint can spread across multiple, noncontiguous BERTs
  - Higher level **$DOG*xxxx*** services encapsulate use of **$PREBERT** and manage storage using **$GETWORK** / **$RETWORK**
- Manages serialization of checkpointed BERTs in the MAS using the **BERT Lock**
- $DOGBERT also provides other services such as checkpoint and query

$JQE display


```
--------------------JES2 Job and Job Output Analysis-------------------------
OPTION ===>                                              SCROLL ===> CSR
Enter JES2 name ===> JES2
                                                              More:     +
Select a specific job and its output by:
   1 Job number or JOBID (decimal)  -->
   2 Job number in hexadecimal      --> 6F20
   3 $JQE address                   -->
   4 $JQE offset from $JOBQPTR      -->
   5 $JQE offset from $CKPTIO       -->
   6 $JQE index                     -->
   7 $JQX address                   -->
   8 $JQX offset from $JQXPTR       -->
   9 $JQX offset from $CKPTIO       -->

Select jobs from multiple queues:
  10 Job queue error analysis
  11 Select all class 'x' jobs      -->

Select all jobs on a specific queue:
  12 Purge queue

Enter UP and DOWN commands to scroll the list of options.
Enter END command to terminate.
```

$JQE display *(continued)*


```
--------------------JES2 Job and Job Output Analysis-------------------------
OPTION ===>                                                SCROLL ===> CSR
Enter JES2 name ===> JES2
                                                                 More:   -
  11 Select all class 'x' jobs      -->

Select all jobs on a specific queue:
  12 Purge queue
  13 Hardcopy queue
  14 Output queue
  15 Receive queue
  16 Setup queue
  17 Class 'x' on execution queue   -->
  18 Xmit queue
  19 Input queue
  20 STC queue
  21 TSU queue
  22 Conversion queue
  23 Spin queue
  24 Free queue
  25 Rebuild queue

Enter UP and DOWN commands to scroll the list of options.
Enter END command to terminate.
```

$JQE display *(continued)*

```
*** $JQE Address=38F08638, Offset=0016C620, Index=00340E
*** $JQX Address=3A3BC360, Offset=000C3348
*** Address of first $BERT for this $JQA is 3E290458
*** BERT lock is not held
*** NOTE:  $JQA incomplete, all fields past label JQABERT are zero
 $JQA:  (Composite of $JQE and $JQX)
          JQE......
    +0000  JQEPRIO.. FF        JQETYPE.. 01        JQEJOBNO. 6F20
    +0004  JQEFLAG1. 00        JQENEXTI. 091C50    JQEFLAG2. 00
    +0009  JQEJOEI.. 000000    JQEFLAG3. 91        JQEJCLAS. A
    +000E  JQEINJNO. 6F20
    +0010  JQEJBKEY. B556485F
    12/19/2044 23:50:05.797733
    +0014  JQETRAK.. 0E05EB01  JQEINPND. 0000      JQEXEQND. 0006
    +001C  JQEFLAG4. 00        JQEDEVID. 000000
    +0020  JQEARMMI. 00        JQEWSLCK. 00
    +0020  JQENEWSU. 0000
    +0022  JQEBUSY.. 00        JQEJLOK.. 00        JQEJNAME. RMT380
    +002C  JQEUSRID. ........  JQESECLB. ........
    +003C  JQENWSID. 00000000
    +003C  JQENWUSE. 00000000
    +003C  JQEJOEID. 00000000  JQEFLAG5. 00        JQEOFFSL. 00
    ...
```

$JQE display *(continued)*


```
...
+0042  JQETGNBR. 0001      JQEFLAG6. 00        JQEFLAG7. 00
+0046  JQEFLAG8. 00        JQEFLAG9. 00
+0048  JQEJBNUM. 00006F20
+0048  JQEARMID. 00        JQEWSLOK. 00        RSV...... 6F20
+004C  JQEDRPRT. 0006017C  JQEDRPRU. ........  JQEDRPUN. 0006017C
+005C  JQESAF... FFFFFFFF
+0060  JQESUMSK. FFFFF800  00000000  00000000  00000000  00000000
+0074            00000000  00000000  00000000
       JQX......
+0080  JQXRECCT. 00000000  JQXMAXRC. 00000000  JQXMXIND. 00
+0085  JQXMAXCC. 000000    JQXBERTT. 00002791
+008C  JQXCRTME. 00000000
09/17/2042 23:53:47.370496
+0090  JQXWSNXT. 00000000  JQXWSPRV. 00000000  JQXJCLAS. A
+00A0  JQXFLAG1. 00        JQXTGWRP. 00        RSV...... 0000
+00A4  JQXNWSID. 00006F20
+00A4  JQXIJNUM. 00006F20
+00A8  RSV...... 00000000  00000000            JQXIT141. 00000000
+00B4  JQXIT142. 00000000  JQXJNUMQ. 00        JQXNJIXI. 000000
       pseudo...
+00BC  JQPTGNBR. 00000000
...
```

$JQE display *(continued)*


```
...
          JQABERT..
+00C0  JQAACCT.. 00000000  00000000  00000000  00000000  00000000
+00D4            00000000  00000000  00000000  00000000  00000000
+00E8            00000000  00000000  00000000  00000000  00000000
+00FC            00000000  00000000  00000000  00000000  00000000
+0110            00000000  00000000  00000000  00000000  00000000
+0124            00000000  00000000  00000000  00000000  00000000
+0138            00000000  00000000  00000000  00000000  00000000
+014C            00000000  00
       JQAXEQ...
+0154  JQAPERF.. 000000    JQAFLAG1. 00        JQASTOK.. 00000000
+015C  JQAWSCN.. ........  JQARRIV.. 00000000  JQAQTIME. 00000000
+016C  JQASID... 0000      JQASCHAF. 00000000  JQASTARM. 00
+0174  JQARHLD.. 00000000  00000000            JQARRSC.. 00000000
+017C            00000000  JQARTOC.. 00000000  00000000
+0180  JQATIMER. 00000000  JQAUTIME. 00000000  JQAFLAG2. 00
+0189  RSV...... 000000
       JQAXBAT..
+018C  JQAXSRMT. 00000000
+0190  JQASCHE.. ................
```

IPCS isn't very smart. OK if displaying from Artificial JQE, but when displaying from checkpoint, all data on this page is zero, even if it isn't

$JOE display


```
--------------------JES2 Job and Job Output Analysis-------------------------
OPTION ===>                                                SCROLL ===> CSR
Enter JES2 name ===> JES2
                                                                 More:     +
Select a specific job and its output by:
   1 Job number or JOBID (decimal)  -->
   2 Job number in hexadecimal      -->
   3 $JQE address                   -->
   4 $JQE offset from $JOBQPTR      -->
   5 $JQE offset from $CKPTIO       -->
   6 $JQE index                     -->
   7 $JQX address                   -->
   8 $JQX offset from $JQXPTR       -->
   9 $JQX offset from $CKPTIO       -->

Select a specific work-JOE and its char-JOE and $JQE by:
  10 Work-JOE index                 -->
  11 Work-JOE offset                -->
  12 Work-JOE address               -->

Select a specific char-JOE and its work-JOEs by:

Enter UP and DOWN commands to scroll the list of options.
Enter END command to terminate.
```

$JOE display *(continued)*


```
----------------------JES2 Job and Job Output Analysis-------------------------
OPTION ===>                                                SCROLL ===> CSR
Enter JES2 name ===> JES2
                                                                 More:   -
Select a specific char-JOE and its work-JOEs by:
  13 Char-JOE index                 -->
  14 Char-JOE offset                -->
  15 Char-JOE address               -->

Select output queues:
  16 Perform output queue error analysis
  17 Held queue
  18 Class 'x' local/userid/remote queues  -->
  19 Class 'x' local queue                 -->
  20 Class 'x' userid queue                -->
  21 Class 'x' remote queue                -->
  22 Network queue
  23 Free queue
  24 Rebuild queue
  25 Char-JOE queue
  26 Purge queue

Enter UP and DOWN commands to scroll the list of options.
Enter END command to terminate.
```

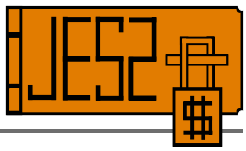
$JOE display *(continued)*


```
The following are from the Held    Queue
$JOE:  Index=00006176 Offset=002797F0 Address=3B2CD808
 $JOE: 3B2CD808
   +0000  ***** WORK JOE *****JOETYPE.. 80        JOENEXTI. 278610
   +0004  JOECURCL. W         JOEPREVI. 00002C    JOEFLAG5. 00
   +0009  RSV.....  000000    JOEFLAG1. 00        JOEJQEI.. 090750
   +0010  JOEFLAG2. 20        JOENXJQI. 000000    JOEFLAG3. 44
   +0015  JOECHARI. 000208    JOEOFFSL. 00        JOECHNXI. 0295C8
   +001C  JOEFLAG4. 00        JOEFLGT2. 00        JOEHOLD.. 00
   +001F  JOEHSRSN. 00        JOEFSID.. 00000000  JOEPRIO.. 0800
   +0026  JOEJNEWL. 0000      JOECPADR. 0829DE09  JOERECCT. 000000C7
   +0030  JOEPGCT.. 00000000  JOEWRECN. 00000000  JOEWPAGN. 00000000
   +003C  JOEIOTTR. 0829DD03  JOEDEVID. 000000    JOEFLAGT. 00
   +0044  JOEROUT.. 00060000  JOERNODE. 0006      JOEREMOT. 0000
   +0048  JOEID.... F1404040  40404040  00010001
   +0048  JOENAME.. 1         JOEID1... 0001      JOEID2... 0001
   +0054  JOECRTME. B9CF5454
   08/02/2003 08:33:07.378959
   +0058  JOECRUID. G017635   JOESWBOT. 00000000  JOEBUSY.. 00
   +0065  JOEFAMLY. 000000
 ...
```

$JOE display *(continued)*

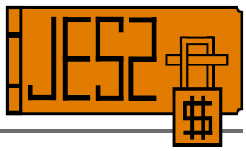

```
$JOE:  Index=00000005 Offset=00000208 Address=3B054220
 $JOE: 3B054220
    +0000  ***** CHAR JOE *****JOETYPE.. 40        JOENEXTI. 004920
    +0004  RSV...... 00        JOEPREVI. 423590    RSV...... 00000000  00
    +000D  JOEWKPTI. 3E14E8    JOEFORM.. STD       JOEFCB... ****
    +001C  JOEUCS... ****      JOEWTRID.           JOEUSER..
    +0030  JOEFLASH. ****      JOEPRMD.. LINE      JOESECLB. ........
    +0044  JOEFLAGC. 00        JOEFLAGD. 00        RSV...... 0000
    +0048  JOEUSE... 0000406D
...
```

Summary



- JES2 libraries, modules, macros, load modules
- Installing JES2 IPCS
- **JES2 base display** which rarely says "the answer is", but provides hints and eliminates suspects
- How JES2 IPCS reports data structure errors
- JES2 module layout
- The ***JES2-Where*** command ($MODLOC)
- Module information from commands and dumps
- Release and subsystem data
- What JES2 start types do
- JES2 address spaces, dataspaces, main task
- When JES2 address space isn't in dump
- System trace and JES2 CTRACE
- JQEs, JQAs, $DOGJQE, and $DOGBERT
- etc.

Bibliography



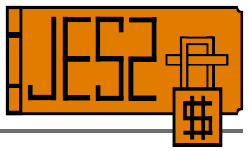
Introduction to JES2 for New System Programmers
SHARE session 2661, John Hutchinson, IBM

JES2 Internals and Exit Overview
SHARE, Winter 2003, Dave Danner, Summit Technical Services

Introduction to JES2 Exit Writing
SHARE, Summer 2001, Chip Wood, IBM

JES2 Table Pairs and $SCANTAB
SHARE, Winter 2003, Chip Wood, IBM

Bibliography *(continued)*



JES2 Diagnosis
GA22-7531-03

JES2 Macros
SA22-7534-03

JES2 Installation Exits
SA22-7536-02

MVS Diagnosis: Reference
GA22-7588-03

MVS Diagnosis: Tools and Service Aids
GA22-7589-03

MVS Diagnosis: Procedures
GA22-7587-00