

JES2 z/OS 1.4 Product Update



SHARE 100, Session 2655

Tuesday, February 25, 2003

Permission is granted to SHARE Inc. to publish
this presentation in the SHARE proceedings. IBM
retains its right to distribute copies of this
presentation to whomever it chooses.

Chip Wood
JES2 Design/Development/Service
Poughkeepsie, NY



chipwood@us.ibm.com

JES2 z/OS releases



- **JES2 z/OS 1.2**
 - Greater than 64K jobs support
 - Dynamic PROCLIB support
 - INCLUDE initialization statement
- **JES2 z/OS 1.4**
 - INCLUDE enhancements
 - JES2 Monitor
 - ▶ **Session 2657, Wed. 11:00**
 - HAM performance/restructure
 - End of Memory cleanup
 - // XMIT JCL
 - Miscellaneous enhancements

JES2 z/OS 1.4 installation



- From **JES2 OS/390 R3** or earlier
 - Migrate to more recent spool-compatible release first (preferably **R8**) to avoid **COLD** start
- From **JES2 OS/390 R4, R5, or R7**
 - Note that **R7** and earlier releases are not supported on **z/OS 1.4 BCP** (*enforced!!!*)
 - **\$ACTIVATE** required to avoid **COLD** start (**R4, R5**)
 - No MAS coexistence (all-member-warm start)
- MAS coexistence from **OS/390 R8-z/OS 1.2**
 - APAR **OW52833** needed on downlevel member
 - **\$ACTIVATE** required on **R8**

JES2

z/OS 1.2

>64K jobs support



- Internal limits increased
 - **JOBDEF JOBNUM** up to **200,000**
 - **JOBDEF RANGE** up to **999,999**
 - **OUTDEF JOENUM** up to **500,000**
 - **CKPTSPACE BERTNUM** up to **500,000**
 - **SPOOLDEF TGSPACE=MAX** up to **16,580,355**
- Limits can now be decreased by operator command
- Job id format changes if new limits exploited
 - **Jnnnnnnn**, **Snnnnnnn**, **Tnnnnnnn**
- Lots of changes to JQE and JOE mappings

Dynamic PROCLIB



- **Problem:** PROCLIBs defined in the JES2 start proc require a JES2 restart to change
 - Change may require ALL MAS members to be restarted
 - Error in JES2 PROC may prevent restart
 - SHARE requirement [SS-JES2-98.203](#)
- **Solution:** Allow dynamic allocation of PROCLIBs
 - **PROCLIB(XXXX)** initialization statement
 - **\$ADD PROCLIB(XXXXXXXX)** command
 - **\$T PROCLIB(XXXXXXXX)** command
 - **\$DEL PROCLIB(XXXXXXXX)** command
 - **\$D PROCLIB(XXXXXXXX)** command

PROCLIB statement



■ New PROCLIB initialization statement

```
PROCLIB (xxxxxxx) DD (n) = (DSNAME=dsn,  
                             VOLSER=volser,  
                             UNIT=unit),  
  
UNCONDITIONAL
```

- ▶ Up to 255 DDs per PROCLIB
 - ▶ VOLSER and UNIT are optional (if cataloged)
 - ▶ UNCONDITIONAL - create even if allocations fail
- ## ■ New operator commands
- **\$ADD PROCLIB(xxxxxxxx)**
 - **\$T PROCLIB(xxxxxxxx)**
 - **\$DEL PROCLIB(xxxxxxxx)**
 - **\$D PROCLIB(xxxxxxxx)**

PROCLIB example



■ Old way (Static PROCLIB)

- ▶ In JES2 PROC:

```
//PROC01 DD DSN=USER.PROCLIB1,VOL=SER=J2COM1,UNIT=3390
// DD DSN=USER.PROCLIB2,VOL=SER=J2COM1,UNIT=3390
// DD DSN=SYS1.PROCLIB
```

■ New way (Dynamic PROCLIB)

- ▶ In JES2 initialization stream

```
PROCLIB(PROC01) DD (1)=(DSN=USER.PROCLIB1,VOLSER=J2COM1,
UNIT=3390),
DD (2)=(DSN=USER.PROCLIB2,VOLSER=J2COM1,
UNIT=3390),
DD (3)=(DSN=SYS1.PROCLIB)
```

- ▶ Modify using \$T PROCLIB(PROC01)

Modifying dynamic proclibs



- To change concatenation for dynamic PROC01
 - Method 1:
 - ▶ **\$T PROCLIB(PROC01),DD(1)=...,DD(2)=...**
 - ▶ Could require several commands due to command length limitations
 - ▶ Advantage: Simplest way if few datasets in concatenation
 - Method 2:
 - ▶ **\$ADD PROCLIB(TEMP01),DD(1)=...**
 - ▶ **\$T PROCLIB(TEMP01),DD(2)=...**
 - ▶ Test and update TEMP01 as required
 - ▶ **\$T PROCLIB(TEMP01),NAME=PROC01**
 - ▶ Advantage: ATOMIC, Allows testing!

Modifying static proclibs



- To change concatenation for static PROC01
 - Method 1:
 - ▶ **\$ADD PROCLIB(PROC01),DD(1)=...,DD(2)=...**
 - ▶ Dynamic definition overrides static definition
 - ▶ **\$T PROCLIB(PROC01)** to update as necessary
 - ▶ **\$DEL PROCLIB(PROC01)** to revert to static definition
 - Method 2:
 - ▶ **\$ADD PROCLIB(TEMP01),DD(1)=...**
 - ▶ **\$T PROCLIB(TEMP01),DD(2)=...**
 - ▶ Test and update TEMP01 as required
 - ▶ **\$T PROCLIB(TEMP01),NAME=PROC01**

\$D PROCLIB example



■ \$D PROCLIB(PROC01)

```

$HASP319 PROCLIB (PROC01)
$HASP319 PROCLIB (PROC01) DD (1) = (DSNAME=USER.PROCLIB1,
$HASP319 VOLSER=J2COM1, UNIT=3390) ,
$HASP319 DD (2) = (DSNAME=USER.PROCLIB2,
$HASP319 VOLSER=J2COM1, UNIT=3390) ,
$HASP319 DD (3) = (SYS1.PROCLIB)
  
```

■ \$D PROCLIB(PROC01),DEBUG

```

$HASP319 PROCLIB (PROC01)
$HASP319 PROCLIB (PROC01) USECOUNT=0, DDNAME=SYS00006,
$HASP319 CREATED=2001.149,20:42:22.36,
$HASP319 DD (1) = (DSNAME=USER.PROCLIB1,
$HASP319 VOLSER=J2COM1, UNIT=3390) ,
$HASP319 DD (2) = (DSNAME=USER.PROCLIB2,
$HASP319 VOLSER=J2COM1, UNIT=3390) ,
$HASP319 DD (3) = (SYS1.PROCLIB)
  
```

- ▶ May also display old concatenations with non-zero use counts, if \$T command has been issued

INCLUDE statement



- **Problem:** Changing JES2 init deck concatenation requires changing JES2 PROC
 - If update is incorrect, JES2 will not start
 - May be difficult to fix when JES2 is down
- **Solution:** New INCLUDE initialization statement
 - Reduces need to update JES2 PROC

INCLUDE statement



■ New **INCLUDE** initialization statement

```
INCLUDE    DSNAME=dsn,  
          VOLSER=volser,  
          UNIT=unit
```

- **DSNAME**=*dsn* may include a member name
- **VOLSER** and **UNIT** are optional (if cataloged)

■ **D INCLUDE**

- Displays current **INCLUDEd** data set
- If not in **INCLUDE**, displays current data set in **HASPPARM** concatenation
- Useful to locate source of error when unexpectedly placed in **CONSOLE** mode due to error in initialization deck

Include statement example



■ Old way:

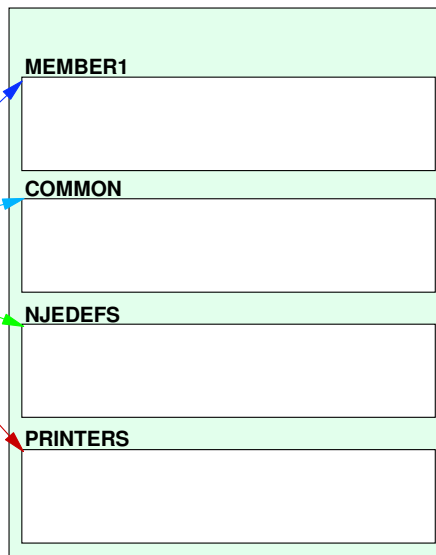
SYS1.PROCLIB

```

//JES2    PROC
//        EXEC PGM=HASJES20,...

//HASPPARM DD DSN=SYS1 . PARMLIB (MEMBER1)
//        DD DSN=SYS1 . PARMLIB (COMMON)
//        DD DSN=SYS1 . PARMLIB (NJEDEFS)
//        DD DSN=SYS1 . PARMLIB (PRINTERS)
    
```

SYS1.PARMLIB



Include statement example

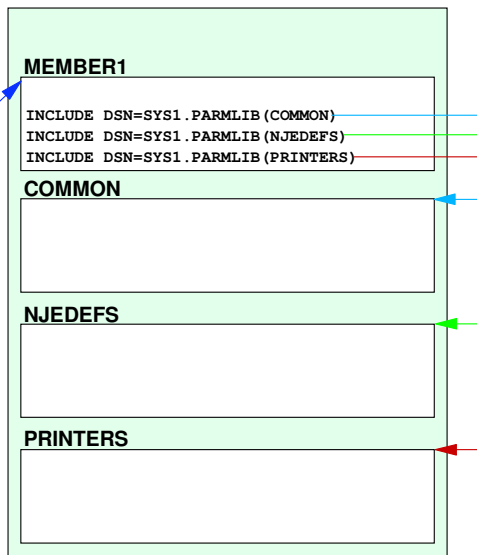


- New way:

SYS1.PROCLIB

```
//JES2   PROC
//      EXEC PGM=HASJES20,...
//HASPPARM DD DSN=SYS1.PARMLIB(MEMBER1)
```

SYS1.PARMLIB



Include statement example

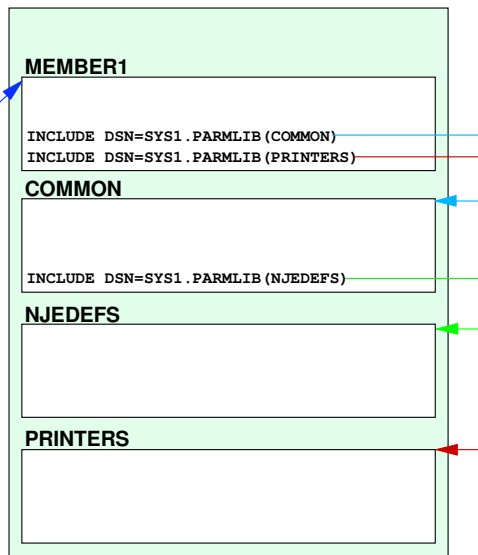


- New way:

SYS1.PROCLIB

```
//JES2 PROC
// EXEC PGM=HASJES20,...
//HASPPARM DD DSN=SYS1.PARMLIB(MEMBER1)
```

SYS1.PARMLIB



Combining INCLUDE and PROCLIB



- Simplify JES2 PROC
 - EXEC, one HASPPARM DD
 - Define PROCLIBS via PROCLIB statement
 - INCLUDE additional HASPPARM datasets
- In emergency, **start JES2 without a PROC!**

S IEESYSAS, PROG=HASJES20, JOBNAME=JES2

- Assumes **HASJES20** in LINKLIST (no STEPLIB)
- When **HASPPARM** open fails, reply to **\$HASP469** message with
 - ▶ **INCLUDE** statement(s) for correct init deck(s)
 - ▶ **PROCLIB** statements (if not in init decks)

JES2

z/OS 1.4

Enhanced INCLUDE Externals



- Updates to INCLUDE initialization statement (new in z2) based on customer input
 - Current syntax of the INCLUDE init statement
 - ▶ **INCLUDE DSNAME=*dsn*,VOLSER=*vol*,UNIT=*unit***
 - New syntax added in z4
 - ▶ **INCLUDE MEMBER=*member***
 - ▶ *member* should be in current parmlib data set being processed
 - ▶ **INCLUDE PARMLIB_MEMBER=*member***
 - ▶ *member* should be in default logical parmlib
 - ▶ Specifying **MEMBER=** in a dataset included by **PARMLIB_MEMBER=** results in logical parmlib search, not just single dataset

Include statement example

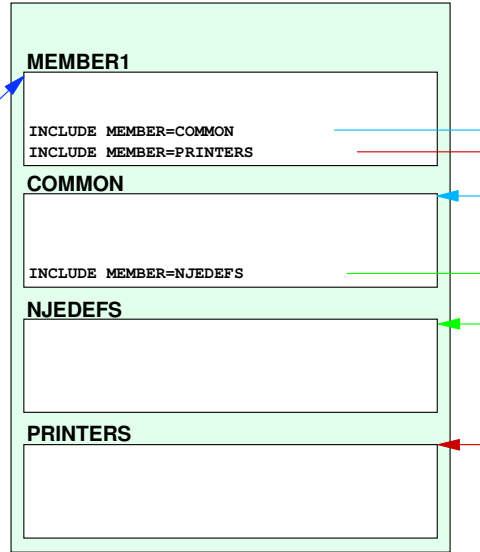


- New way:

SYS1.PROCLIB

```
//JES2   PROC
//      EXEC PGM=HASJES20,...
//HASPPARM DD DSN=SYS1.PARMLIB(MEMBER1)
```

SYS1.PARMLIB



Default Parmlib Member



- New start option (to read from default PARMLIB)
 - ▶ **S JES2,PARM=**(**'MEMBER=member'**) or
 - ▶ **S JES2,PARM=**(**'PARMLIB_MEMBER=member'**)
 - ▶ Reply **MEMBER=member** to **\$HASP467**
 - ▶ *member* should be the member of logical parmlib in each of these cases
- **HASPPARM=ddname** and **MEMBER=** are mutually exclusive
- If neither **HASPPARM=** nor **MEMBER=** is specified, then it will process from the default **HASjesx** member of logical parmlib (*jesx* is subsystem name)
 - ▶ IBM does not ship a default parmlib member

Search order



- If **HASPPARM=ddname** parameter is specified, use that DD
- If **MEMBER=/PARMLIB_MEMBER=** parameter is specified, use that member from logical parmlib
- If neither is specified
 - Try to open DD with ddname of **HASPPARM**
 - If **HASPPARM** DD is not found, use **HASjesx** member from logical parmlib

Include statement example

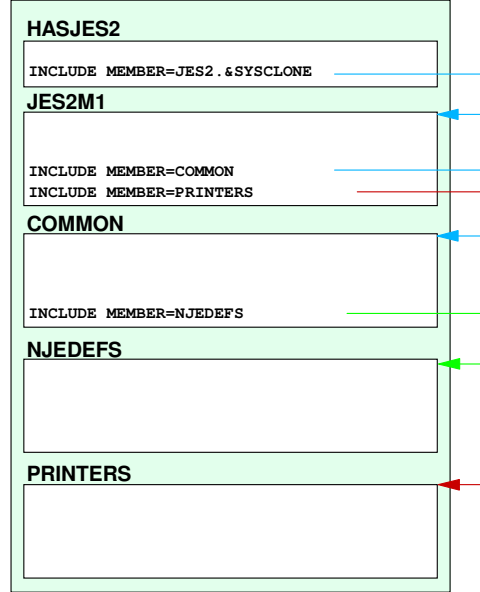


- New way:

SYS1.PROCLIB

```
//JES2    PROC
//        EXEC PGM=HASJES20,...
```

SYS1.PARMLIB



Starting JES2 without a JES2 PROC



- In emergency, **start JES2 without a PROC!**

```
S IEESYSAS, PROG=HASJES20, JOBNAME=JES2
```

- Assumes HASJES20 in LINKLIST (no STEPLIB)
- When **HASPPARM** open fails, logical parmlib member **HASJES2** will be used
- If **HASJES2** not found, **\$HASP469** issued

```
S IEESYSAS, PROG=HASJES20, JOBNAME=JES2,  
  PARM=' MEMBER=MEMBER2 '
```

- Uses logical parmlib member **MEMBER2** (no open of **HASPPARM DD** attempted)
- If **MEMBER2** not found, **\$HASP469** issued

JES2 Monitor



- JES2 "Health" Monitor
 - 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 **jesxMON** (**jesx** - subsystem name)
 - Samples JES2 address space to determine current status of JES2 main task and resource utilization
 - Anomalies surfaced as "alerts" if not resolved in a few seconds (MVS WAITs, long running PCEs, etc)
 - **\$Jxxx** commands to display when JES2 is sick
- **Session 2657, Wednesday 11:00**

HAM I/O Improvements



■ Problem:

- HAM I/O is single record (except FSS)
- HAM code is old with RAS problems

■ Solution:

- Use multi record I/O to read and write SYSOUT and SYSIN data from/to SPOOL
- Use EXCPVR to reduce overhead
- Improve overall RAS
- (Almost) No external changes
- Massive internal changes

HAM internal changes



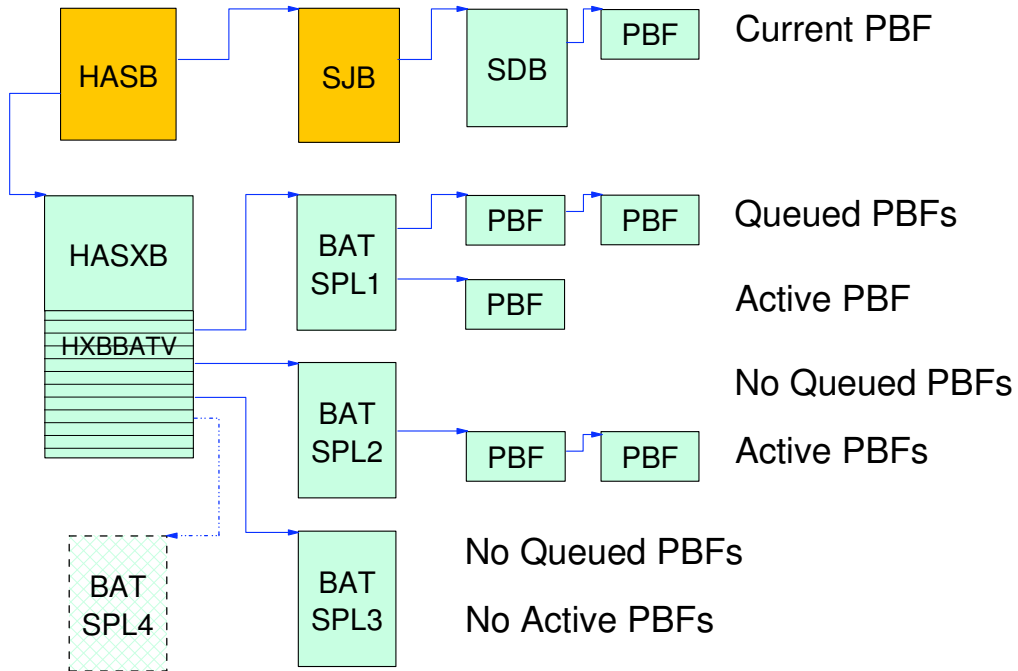
- Internal reader processing was unchanged
- SVC 111 only used for internal reader
 - PC routines used in all other cases
- New SDB lock to serialize all PC services
- EXCPVR used instead of EXCP
- Format 1 CCWs used (31 bit CCW)
- I/O associated with job step TCB not current
 - Less code needed at EOT
- SDBs are above the line
 - DEB pointer to SDB is 31 bit address shifted 7 bits
 - SDBs are obtained on appropriate boundary

HAM PUT internal changes



- HAMPUT does a PC for every record
 - No UBUF for PUT
- PUT can be from any key
- Serialized by new SDB lock
- Counts records as they are PUT
 - Not when buffer is full
 - May affect excession exit (\$EXIT 9)
- PBUFs for PUT queued by volume not data set
- 1 EXCPVR can write up to 12 buffers

Data Structure (PUT)



Data Areas (PUT)



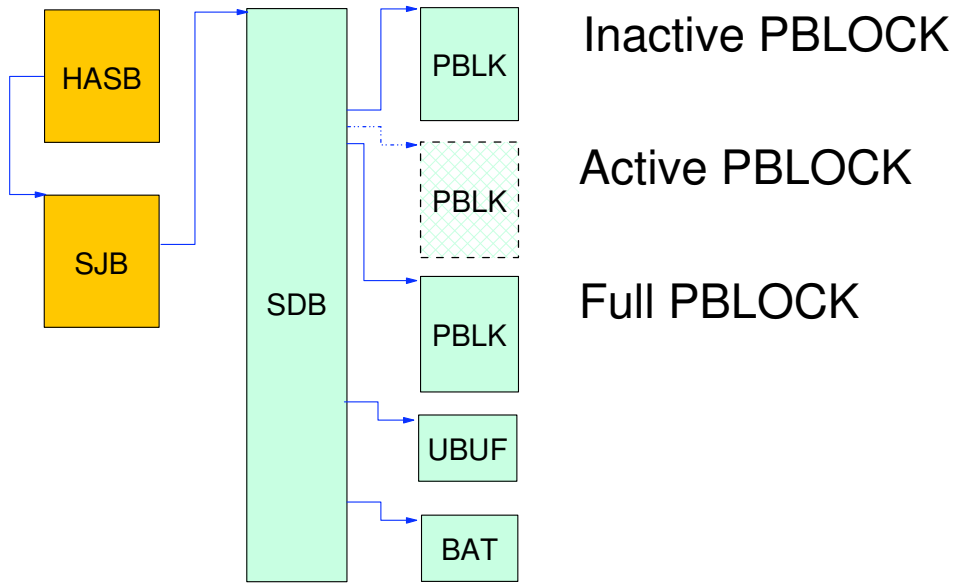
- SDB - Subsystem data set block
 - One active PBF, no UBF
 - Full PBF gets queued to BAT for volume
- HXBBATV - BAT vector in the HASXB
 - One per potential SPOOL volume
 - BATs obtained as needed (for writes)
 - BATs are not freed until HASXB freed
- BAT - Buffer Auxiliary Table
 - Below the line IOB
 - Pending and active PBF chains

HAM GET internal changes



- HAMGET still uses a UBUF
 - PBUFs are no longer used
 - PBLKs are used instead
- PC when UBUF empty
- FSS and normal GETs all work the same
 - Track cell (and more) reads for all GETs

Data Structure (GET)



Data Areas (GET)

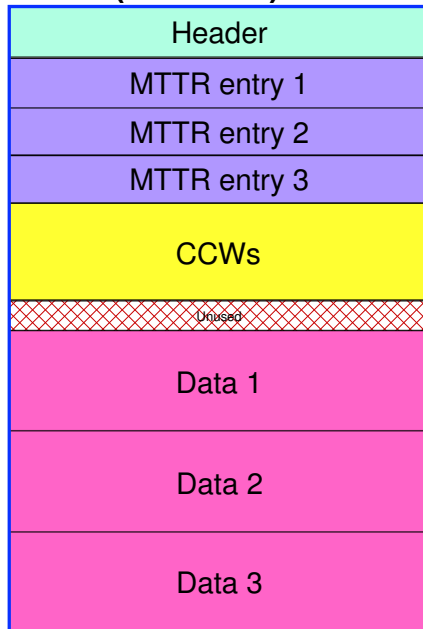


- SDB - Subsystem data set block
 - Three queues of PBLKs
 - ▶ Inactive PBLKs (empty, no I/O active)
 - ▶ Active PBLKs (I/O active)
 - ▶ Full PBLKs (I/O completed, Buffers full)
 - 2 buffers obtained when SDB obtained
- BAT - Buffer Auxiliary Table
 - One per SDB
 - IOB used for I/O

Data Structure (PBLK)



PBLK (3 MTTRs)



- Always multiple of pages
- Data contiguous at end
- No headers in front of data
- Can have 1-12 MTTRs
 - ▶ 12 is one track
- Could be no unused space
- Entire PBLK read in one EXCPVR
- Can have unused entries
 - ▶ Short track group

Data Area (PBLK)



- PBLK - Protected Block
 - Buffer to read HDBs into
 - Size set at data set open
 - ▶ Non-trackcelled, 1 buffer
 - ▶ Trackcelled, TRKCELL buffers
 - ▶ SPIN, SYSIN, 1 tracks worth of buffers (3390)
 - MTTR entries correspond to buffers, contains
 - ▶ MTTR and address of buffer
 - ▶ Flags
 - ▶ IDAWs

Blank Truncation



- "Null" records no longer changed to "blank" records by HAM processing
 - No longer pad with single blank
 - Could affect line counts for some datasets
 - Consistent with NJE processing
- Blank truncation on FSS printers
 - Original LRECL now passed over the FSS interface
 - Allows FSS printers to pad out blanks that were deleted by JES2
 - Requires FSS exploitation
- New field in IAZIDX
 - **IDXORECL** - 2 byte original LRECL
 - Only in first in segment of record

\$EXIT 9



- Records are counted for each PUT, not each full buffer
- Exit 9 gets control at exact excession count, not first full buffer after excession
- "Should" not affect most exits

Count on entry (pre-z4)	Count on entry (z4)
4039	4001
8067	8001
12095	12001
16017	16001
20045	20001
24073	24001
28101	28001
32023	32001
36051	36001

End of Memory (EOM)



■ **Problem:**

- The BCP began terminating EOM TCBs in z2 if they took too long to finish without making progress (abend 30D)
- JES2's EOM activities entailed WAITing for the JES2 main task.
- The JES2 main task could be down (JES2 abend) or awaiting access to the checkpoint
- JES2 EOM could be abended resulting in lost (JES2) resources

■ **Solution:**

- Don't wait for JES2 main task from EOM processing

End of Memory (EOM)



- Solution details
 - Use the JES2 SJB (Subsystem Job Block) to represent an executing unit of work, but nothing else
 - ▶ PSO control block moved to dataspace (*jesxPSO*) and used for communication with JES2 for PSO requests
 - ▶ New STAC control block in dataspace (*jesxSTAC*) used for communication with JES2 for status/cancel requests
 - ▶ Do not wait for the JES2 address space to update JQEs before allowing MVS to reuse the terminating address space
 - ▶ Do not wait for JES2 to finish Internal Reader Processing

EOM Line item side effect



- JES2 no longer checks TSO logons for duplicate logons. The same userid can be logged on more than once in a JESplex.
- Installation controls this by making **SYSIKJUA** be a **SYSTEMS** ENQ
 - New routine in sample exit **HASX44A** can be enabled to restore checking to HASPCNVT
- Multiple instances of a single userid logged onto a JESplex is **not officially supported**.

EOM Line item side effect (*continued*)



- What will happen if multiple logons of same userid?
 - TSO GR (Generic Resource) will fail to determine what MVS should be used for reconnect.
 - An MCS **SEND** command to a given user will work if user is logged onto same MVS image as SEND command. The message will go only to the "local" instance
 - An MCS **SEND** command will not go to any of the multiple instances if none are logged onto the "local" MVS image
 - Notify messages will only be sent to first instance found

XMIT JCL card



- JES2 now supports the `// XMIT` JCL card to route job execution to another node.
 - Previously only supported by JES3
 - Requirement [SO-JES2-93.006](#)
 - The **SUBCHARS=** keyword is not supported by JES2 (JCL error)
 - `/*XMIT`, `/*XEQ` and `/*ROUTE XEQ` still supported

Invalid Jobname Character



- Invalid JOBNAME character now specifiable
 - Requirement [SO-JES2-97.203](#)
 - Was always a ?
 - ▶ For displays, treated as generic
 - ▶ **\$DJOBQ(*?*)** does not display all jobs with a bad jobname character
 - **JOBDEF BAD_JOBNAME_CHAR=**
 - ▶ Can now be ? / + : _ - ! or alpha/numeric/special
 - ▶ Only affects newly arriving jobs
 - Example
 - ▶ **JOBDEF BAD_JOBNAME_CHAR=!**
 - ▶ **\$DJOBQ(*!*)** displays all jobs with an !

\$TRACE 32



■ \$TRACE 32 - \$#REM

- Traces every \$#REM call
- Use to determine why output is disappearing unexpectedly
- Dumps JOE contents in hexadecimal and EBCDIC

```

14.47.51.42 ID = 32 $#REM      JOB00009  COMM      06B7B3C8  JOE REMOVAL
   0  80000000 C1000006 00000000 00000005 20000000 48000005 00000000 02000000 *...A..
  20  00000000 09000000 0C001005 0000001D 00000000 00000000 00000000 0B001004 *.....
  40  00000000 00010000 F1404040 40404040 00010001 B7F86252 4E4E4E4E 4E4E4E4E *.....
  60  00000000 00000000

```

\$TRACE 33



- **\$TRACE 33** - NJE Header/Trailer
 - Traces every NJE Header or Trailer or NMR
 - No need to decode \$TRACE 4 or 5
 - Specify **TRACE=YES** on **OFFLOAD**, **LINE**, or **NODE**

```

14.58.14.14 ID = 33 NJEHDR STC00011 L19.ST1 06B17180 TRANSMIT JOB HEADER
JOB HEADER PREFIX
      NJHLEN=0170 NJHFLAGS=00 NJHSEQ=00
JOB HEADER GENERAL SECTION
      JOBNAME=DEALLOC JOB NUMBER=11 ORIGIN=NODE1 """""""" EXECUTION=NODE1
0 00D40000 000BD0C1 400F0F01 00000000 00000000 00000000 C4C5C1D3 D3D6C340 *.M...
20 00000000 00000000 00000000 00000000 00000000 00000000 B7F8646C A62E3245 *.....
40 D5D6C4C5 F1404040 00000000 00000000 D5D6C4C5 F1404040 40404040 40404040 *NODE1
60 D5D6C4C5 F1404040 40404040 40404040 D5D6C4C5 F1404040 40404040 40404040 *NODE1
80 E2E3C440 40404040 00000002 00015180 3B9AC618 000F423F 40404040 40404040 *STD
A0 40404040 40404040 40404040 40404040 40404040 40404040 40404040 40404040 *
C0 40404040 0000001D 0000000B 00000000 00000000 * ..
JES2 SECTION OF THE JOB HEADER
0 00348400 01000000 00000000 00000000 00000000 00000000 00000000 00000000 *..d...
20 00000000 00000000 00000000 00000000 00000000 *.....
JOB SCHEDULING SECTION OF THE JOB HEADER
0 000C8A00 000F423F 7FFFD78 *.....
SECURITY SECTION OF THE JOB HEADER
0 00588C00 00048000 50012006 C0010000 00000000 00000000 D5D6C4C5 F1404040 *.....
20 00000000 00000000 D5D6C4C5 F1404040 00000000 00000000 E2E3C3C9 D5D9C4D9 *.....
40 00000000 00000000 4E4E4E4E 4E4E4E4E 00000000 00000000 *.....
    
```

Miscellaneous changes



- **INITDEF PARTNUM=0** now valid
 - No JES initiators defined
- JES2 now passes a count of jobs that can run only on constrained systems in WLM sampling data.
 - WLM can better decide whether to start initiators on constrained systems
 - **\$DPERFDATA(SAMPDATA)** displays count

```

$HASP660 $DPERFDATA(SAMPDATA)
$HASP660 SERVICE CLASSES KNOWN TO JES2:
$HASP660 SRVCLASS(WLMSHORT)=(TOKEN=16748000,REGISTERED,SYSTEMS=
$HASP660 (AQFT,AQTS))
$HASP660 SERVICE CLASS SAMPLING DATA:
$HASP660 SRVCLASS(22)=(SYS_QUEUE=0,SYS_INEL=9,SYS_LIMIT=0,
$HASP660 LOCAL_QUEUE=0,CONSTRAINT_AFFINITY=0,LOCAL_INEL=9)
$HASP660 REPORT CLASS SAMPLING DATA:

```