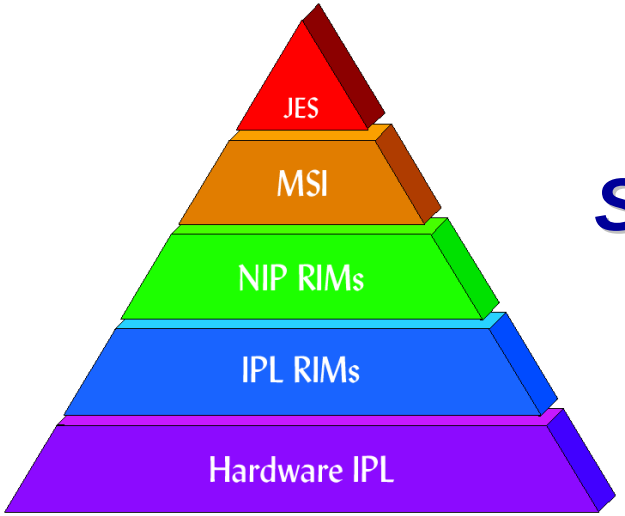


z/OS MVS
System Initialization Logic
Initial Program Load (IPL)

Silvio Sasso
IBM Switzerland, Global Services ITS
sisa@ch.ibm.com

Version 2, July 2014

Trademarks and Disclaimers

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

AIX*	System z*
DB2*	zEnterprise
IBM*	z/OS*
IBM (logo)	z/VM*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license there from.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
InfiniBand is a trademark and service mark of the InfiniBand Trade Association.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.
* All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

zEnterprise Disclaimer

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

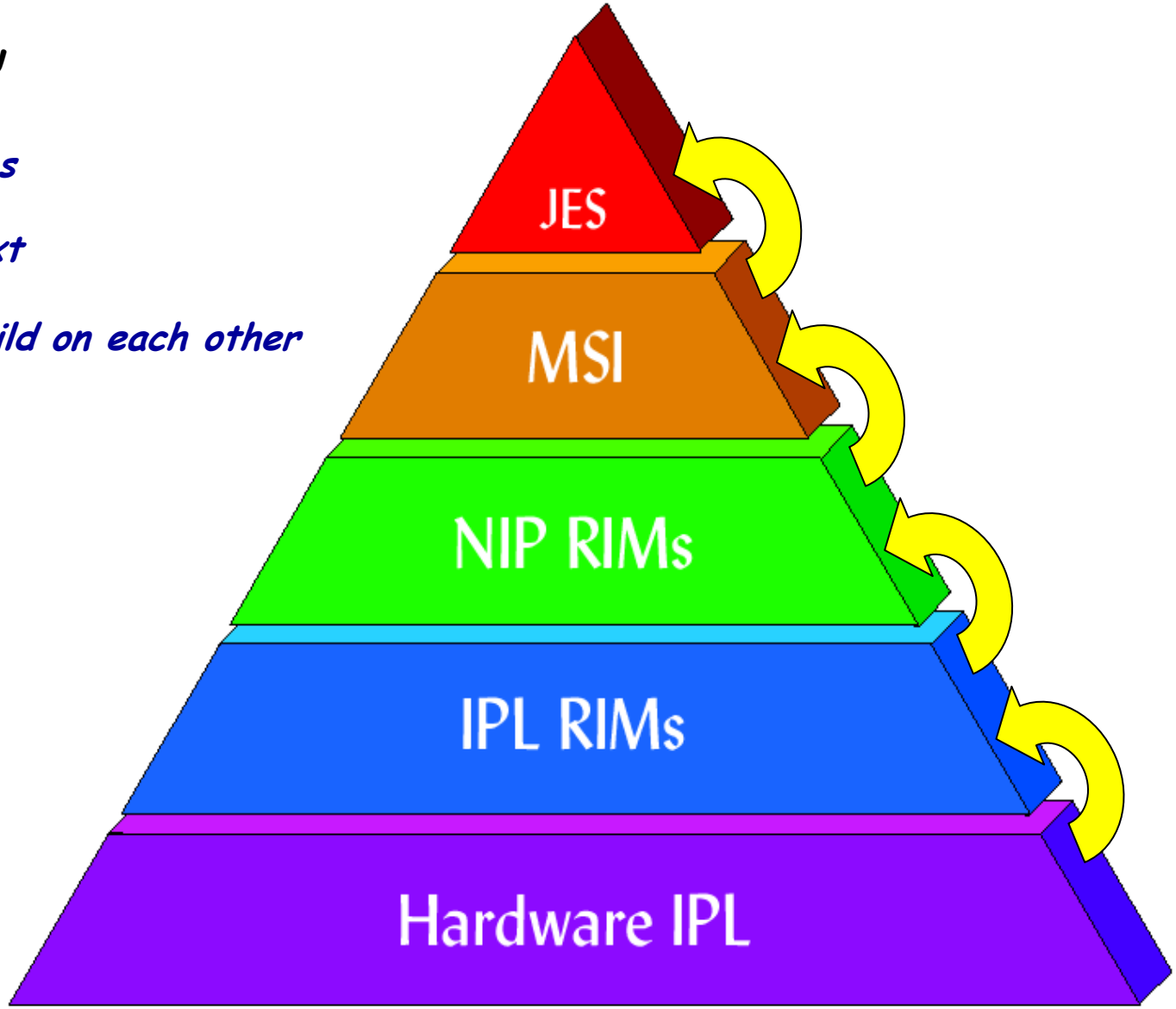
- *This presentation will describe the general processing which is involved in initializing a z/OS system, from the IPL process until the system is ready to start either JES2 or JES3*
- *The major steps described are:*
  - *The hardware process of loading z/OS*
  - *The loading and initialization of the nucleus*
  - *The initialization of general system resources*
  - *Master Scheduler Initialization*
- *In addition, this presentation will provide you with information on how to create an IPL statistics report:*
  - *From an SVC dump using IPCS*
  - *Using the IPLSTATX and IPLSTATZ utilities provided on the z/OS USS tools and toys website*

Special Notices

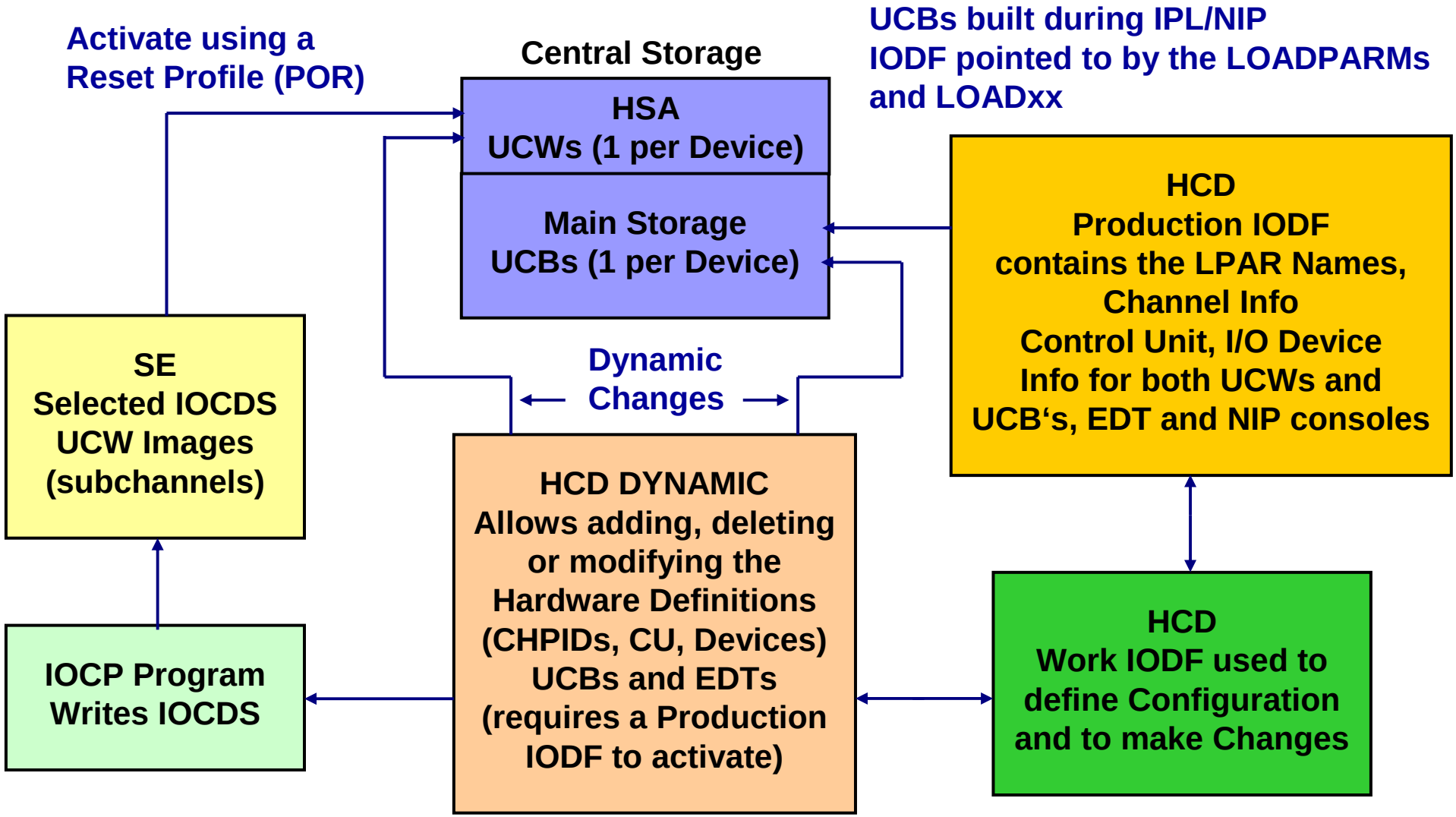
- *This documentation contains some parts of the presentation "Pulling a System up by its Bootstraps" by Greg Dyck (IBM), SHARE session 2854, March 2000*
- *IBM may change the implementation of internal processing at any time with no prior notice*

z/OS Initialization Overview

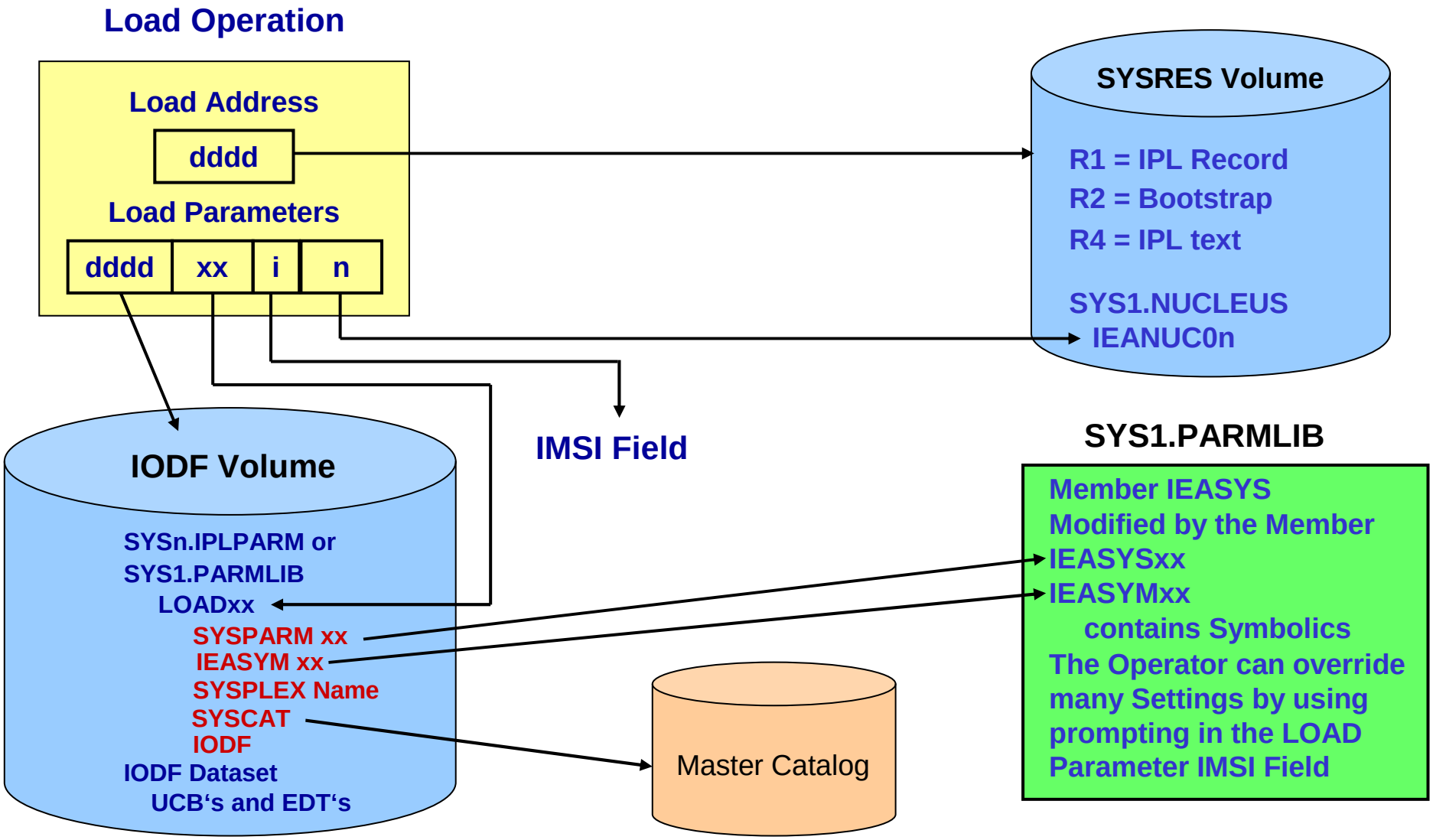
- *Processed in different phases*
- *Each phase builds on the next*
- *Within each phase, steps build on each other*



UCW and UCB Generation



Dataset Considerations – the big Picture



Load Parameters

IODF dddd	LOADxx xx	IMSI i	NUCx n

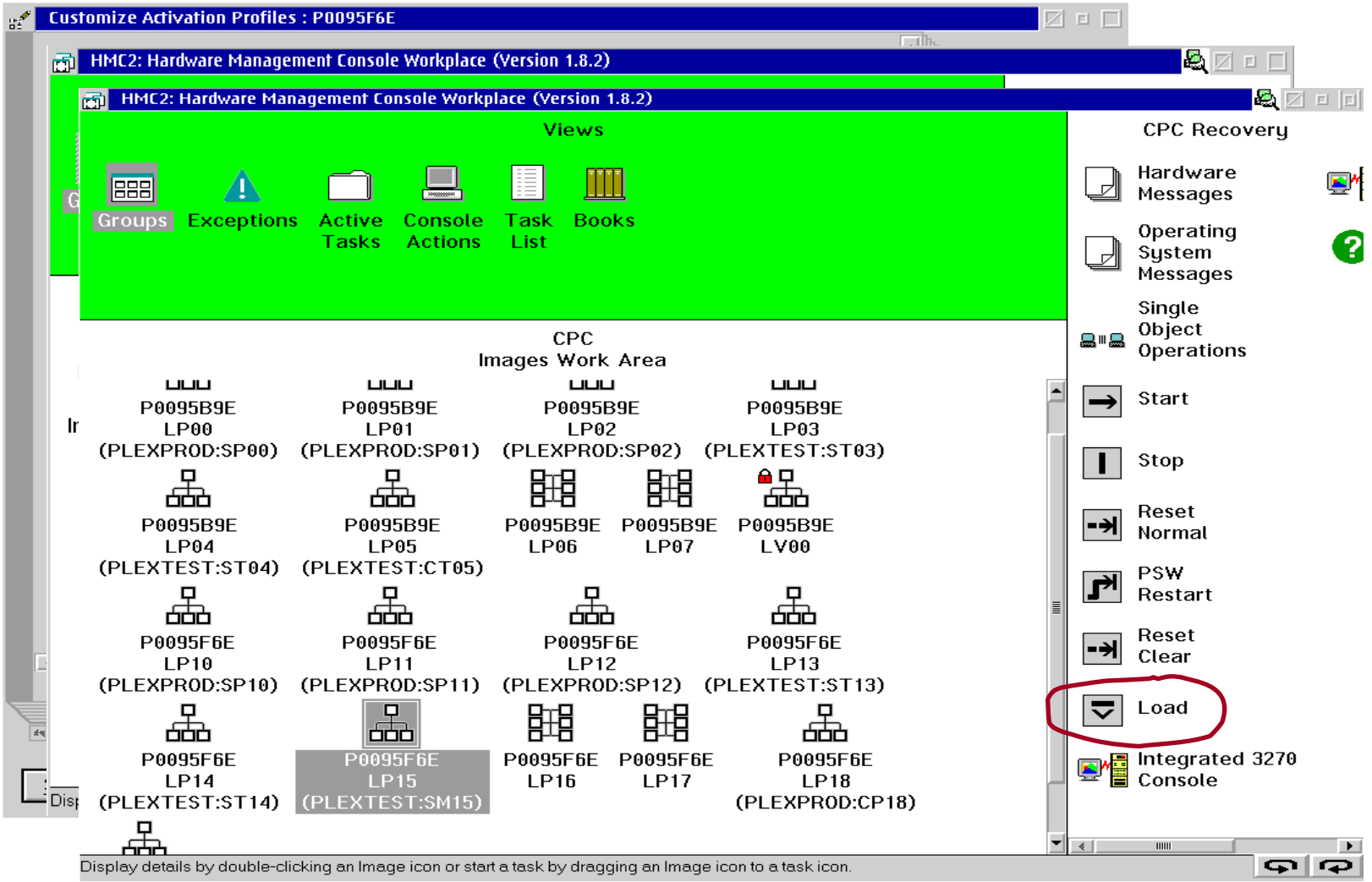
- *DDDXXINN Load Parameter Values*
  - *DDDD:* *Device number of the volume containing the IODF dataset (Default is SYSRES)*
  - *XX:* *ID of the LOADxx member to be used (the default is LOAD00)*
  - *I:* *Initial Message Suppression Indicator (IMSI)*

    The default suppresses most informational messages and does not prompt for system parameters; will use the LOADxx values
  - *N:* *Nucleus ID to be used (default is 1: IEANUC01)*

IMSI Character

IMSI Character	Display informational Messages	Prompt for Master Catalog Response	Prompt for System Parameter Response
Period (.) or blank	No	No	No
A	Yes	Yes	Yes
C	No	Yes	No
D	Yes	Yes	No
M	Yes	No	No
P	No	Yes	Yes
S	No	No	Yes
T	Yes	No	Yes

And all begins with a Mouse Click…



Hardware IPL Overview

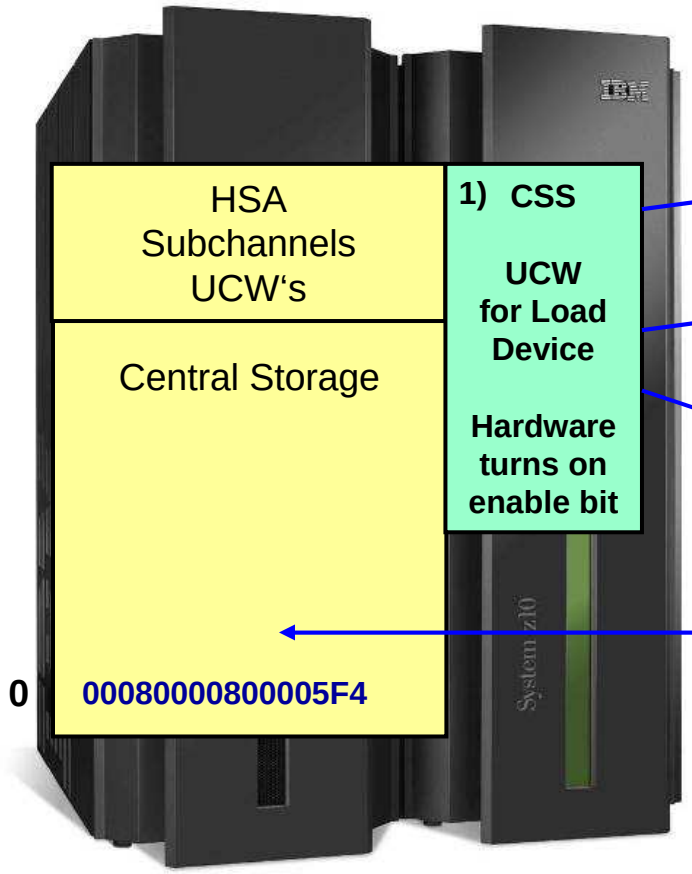
- *Process is defined by the z/Architecture*
- *Controlled by hardware*
- *A single CPU is used for IPL - all other CPUs are placed into a manual (i.e. stopped) state*
- *A hardware system reset occurs before the process begins*
- *IPL records are provided in SYS1.SAMPLIB and written with ICKDSF*
  - *Cyl 0, Trk 0, R1, R2, IEAIPL00*
- *Sample JCL to write IPLTEXT to a SYSRES Volume:*

```
//INSTTXT EXEC PGM=ICKDSF
//SYSPRINT DD  SYSOUT=*
//IPLVOL   DD  DISP=OLD,UNIT=SYSDA,VOL=SER=vvvvvv
//IPLTEXT  DD  DSN=SYS1.SAMPLIB(IPLRECS),DISP=SHR,
//             UNIT=SYSDA,VOL=SER=vvvvvv
//         DD  DSN=SYS1.SAMPLIB(IEAIPL00),DISP=SHR,
//             UNIT=SYSDA,VOL=SER=vvvvvv
//SYSIN    DD  *
   REFORMAT  DDNAME(IPLVOL)        -
             IPLDD(IPLTEXT,OBJ)    -
             NOVERIFY              -
             BOOTSTRAP
//
```

Volser of IPL (SYSRES) Volume

Hardware IPL Flow

Load Operation



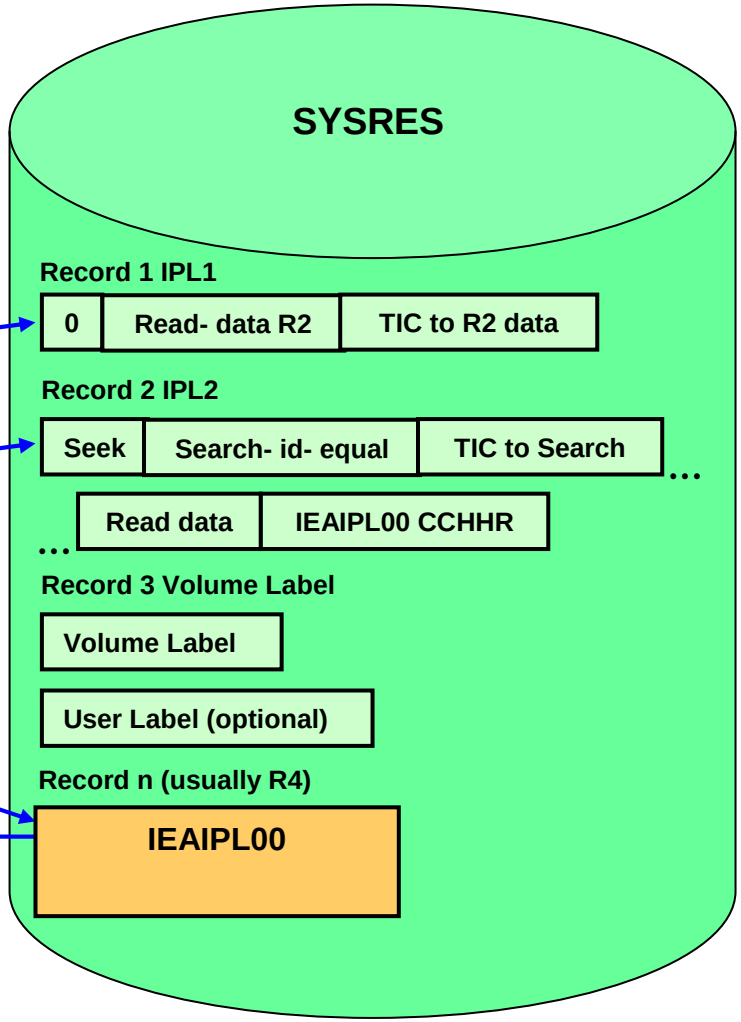
2) CSS stores IPL CCW into absolute Storage Location 0 and initiates SSCH with CCW Addr = 0

DASD will then seek to CYL 0, Head 0 and read Record 1

3) Record 1 will provide CCW's to read Record 2

4) Record 2 will provide CCW's to read Record 4

5) IEAIPL00 Location 0 contains initial PSW



After Record 4 has been read the Hardware Portion of IPL is complete

Hardware IPL Summary

- *Hardware generates an IPL CCW to read of 24 bytes IPL text into location 0*
  - *CCW = 02000000,40000018*
  - *For DASD, this always reads cylinder 0, track 0, record 1*
- *Location 8 treated as a command chained CCW*
  - *Read record 2 into storage, command chain to next CCW*
  - *Transfer CCW execution to record 2 location*
  - *Seek and search for IEAIPL00 record*
  - *Read IEAIPL00 into location 0*
- *CCW chain completion, PSW is loaded from absolute 0 and execution begun*
  - *IEAIPL00 location 0 contains initial PSW*

- **Overview**
  - *Originally just loaded the Nucleus and set up the Master address space environment*
    - *Processing has gotten more complex with the XA architecture and Dynamic I/O support*
  - *Processing is single threaded*
  - *The IPL vector table (IVT) contains global information during this phase*
- **IEAIPL00**
  - *A mini operating system - non relocatable*
  - *Builds an initial virtual environment*
    - *IPL workspace located at X'20000000' virtual*
  - *Provides services to*
    - *Back virtual storage with real frames*
    - *Do I/O*
  - *Controls the IPL initialization process*
    - *Loads IPL Resource Initialization Modules (RIMs) into workspace*
    - *Gives them control*

IPL RIM Processing

1. *Test Block Instruction (clear Storage)*
2. *Read SCPINFO*
   - *Get loadparm*
   - *Set autostore status on**
3. *Locate usable real storage at top of memory*
4. *Get IPL load parameters, and set any defaults*
5. *Search LOADxx, process the information in LOADxx*

```
IEA371I SYS0.IPLPARM ON DEVICE 5411 SELECTED FOR IPL PARAMETERS ← first Message displayed on NIP Console
IEA246I LOAD   ID 00 SELECTED
```

6. *Search IODF, process the information in the IODF*

```
IEA246I NUCLST ID 00 SELECTED
IEA519I IODF DSN = SYSIOD.IODF24
IEA520I CONFIGURATION ID = SM15DPRI. IODF DEVICE NUMBER = 5411
```

- *Build a table of NIP consoles*
  - *max. number of NIP consoles supported by IPL RIM is 64 (HCD supports 128)*
    - *see APAR OA12877 for additional information*

***Store Status**

The store-status operation places an architectural-mode identification and the contents of the CPU registers, except for the TOD clock, in assigned storage locations.

For more information, refer to „zArchitecture Priciples of Operations“, SA22-7832

IPL RIM Processing…

6. *process the information in the IODF (cont.)*
   - *Invoke the device UIMs to*
     - *Identify device specific nucleus and LPA modules*
     - *Calculate required SQA and ESQA*
     - *Build device control blocks in the workspace*
     - *Build the Allocation EDT in the workspace*
7. *Create a map of the DAT-on nucleus CSECTs*

```
IEA091I NUCLEUS 1 SELECTED
IEA093I MODULE IEANUC01 CONTAINS UNRESOLVED WEAK EXTERNAL REFERENCE
IFFIOM
IEA093I MODULE IEANUC01 CONTAINS UNRESOLVED WEAK EXTERNAL REFERENCE
IEDQATTN
IEA093I MODULE IEANUC01 CONTAINS UNRESOLVED WEAK EXTERNAL REFERENCE
IECTATEN
```

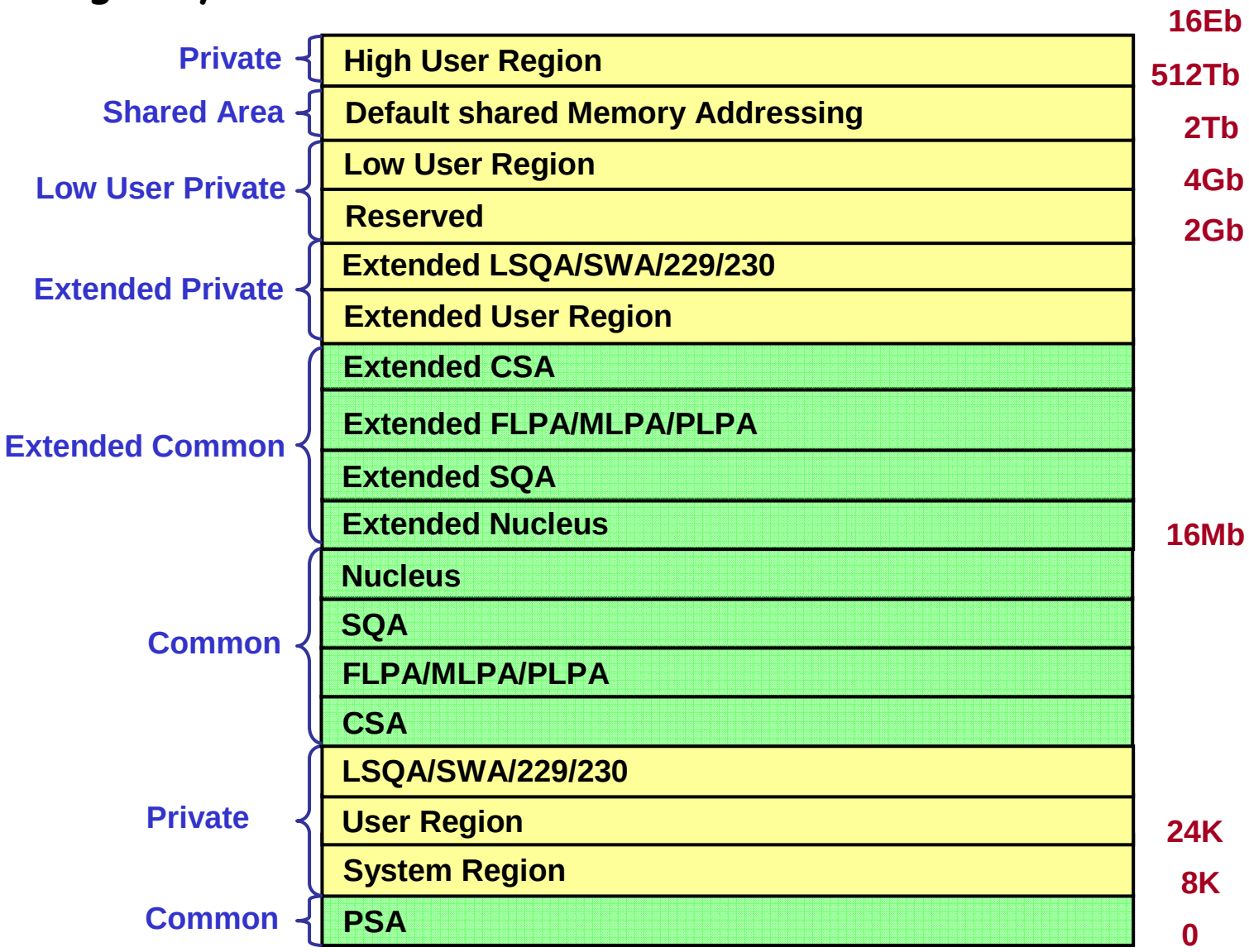
   - *Includes modules contained in IEANUC0x and IEANUC2x, and those identified by NMLs, NUCLSTxx, and UIMs*
   - *CSECTs are grouped/positioned by attributes, RMODE and read-only*
8. *Load modules, dynamically resolving external references*

IPL RIM Processing…

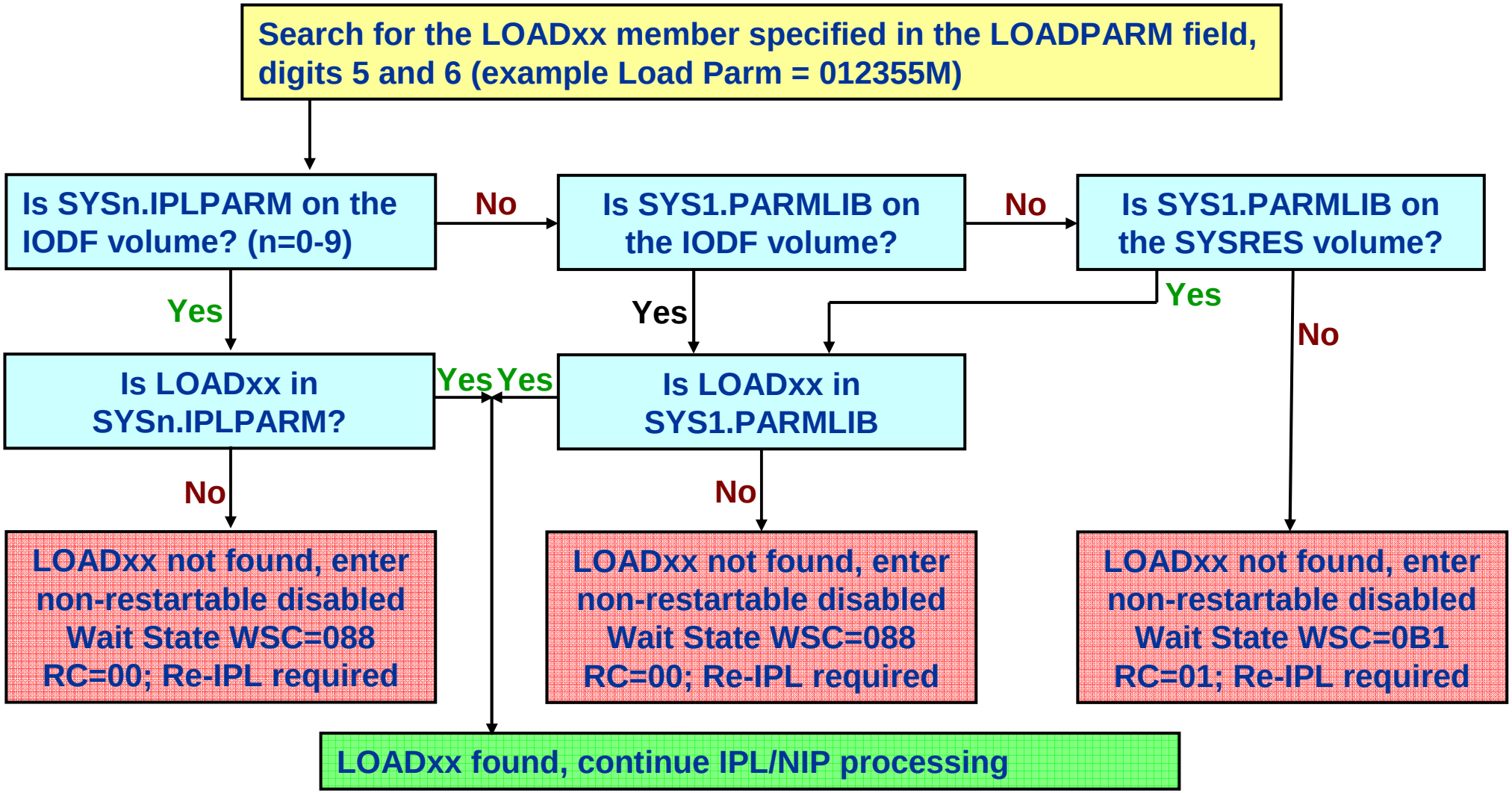
9. Create the initial SQA/ESQA areas
   - Sum of IBM supplied value, LOADxx INITSQA, UIM determined value
10. Create Master's VSM control blocks and LSQA
11. Create Master's permanent page and segment tables
12. Move from the workspace into SQA/ESQA
    - Device control blocks
    - Allocation EDT
    - IPL Messages
    - LPA device support module list
13. Validate real storage, build available frame queue
    - IPL workspace is destroyed
14. Load Prefix Register
15. Switch to nucleus version of the PSA

Note: this is just a brief overview of the IPL RIMs. For a complete list of all IPL RIMs refer to the IPCS IPL statistics report at the end of this presentation

Virtual Storage Layout

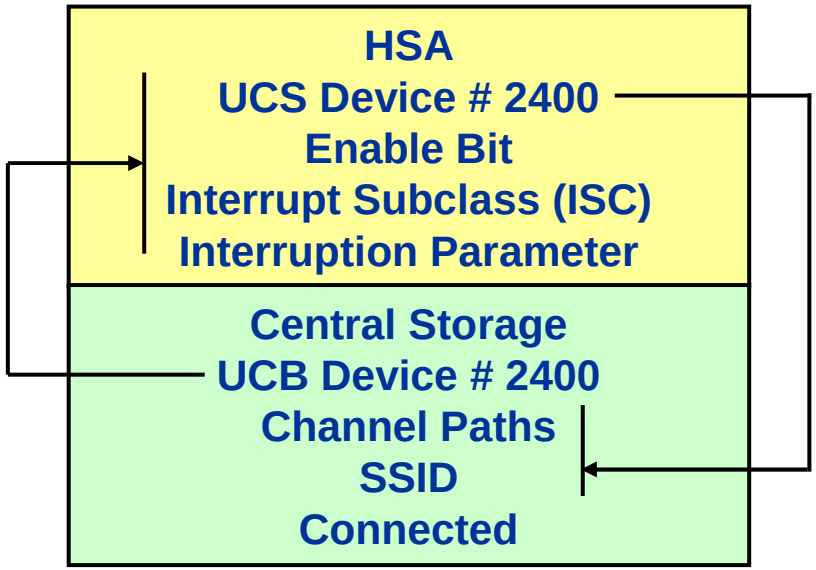


LOADxx Search Sequence



- **Overview**
  - *Initializes basic system resources*
  - *Processing is multithreaded - normal dispatching of work is done*
  - *Basic system service (SRBs, WAIT, POST, EXCP, ATTACH, etc.) are initially available*
  - *Additional services enabled as NIP RIMs run*
  - *The NIP vector table (NVT) contains global information during this phase*
- **Control routine**
  - *Sets traps for unexpected errors (no RTM support is available yet)*
  - *Verifies the hardware environment*
  - *Creates IPL processor control blocks*
  - *Creates global VSM control blocks*
  - *Creates I/O control block pools*
  - *Creates the initial system trace table*
  - *Opens SYS1.NUCLEUS as the LNKLST*
  - *Loads and invokes NIP RIM routines*

UCW to UCB Mapping



In order for MVS to use a device:

- ***a UCW for the device must exist***
- ***a UCB for the device must exist***

During device mapping:

- ***each matching UCW is enabled***
- ***each matching UCB is connected***

During the mapping process, the I/O configuration (UCWs) loaded into the HSA with a POR (or updated via dynamic I/O) is matched with the operating system configuration (UCBs) defined in the IODF

The UCWs are placed in the *disabled* state after POR or system reset

Initial UCB state:

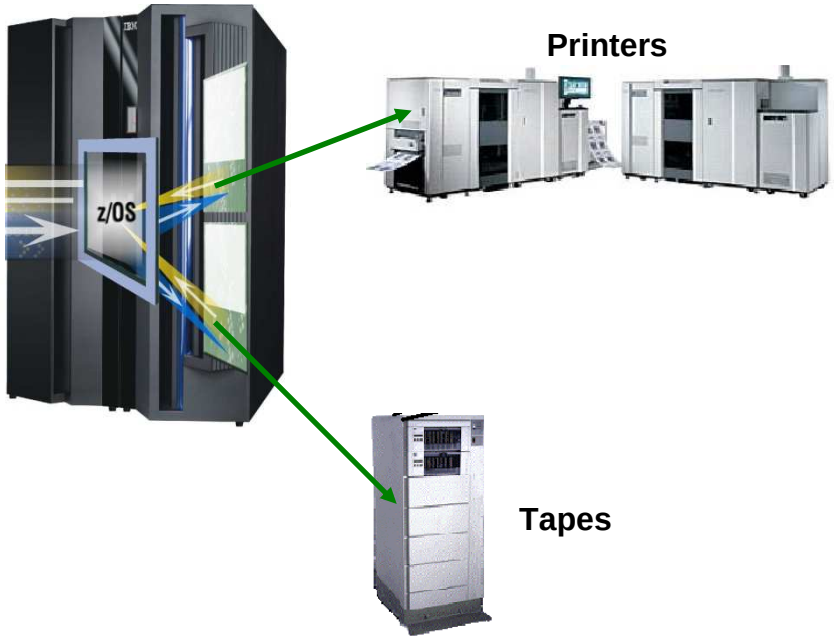
- **the UCBs are built with the "not connected" state bit = 1 (UCB byte 7, bit 2)**
- **at the completion of this mapping process all devices defined to both the channel subsystem (UCWs) and MVS (UCBs) will be enabled and connected**
  - ***any UCWs without corresponding UCBs will be left disabled***
  - ***any UCBs without corresponding UCWs will be left not connected***

Devices in either one of these states cannot be used by the system

Non-DASD Pathing

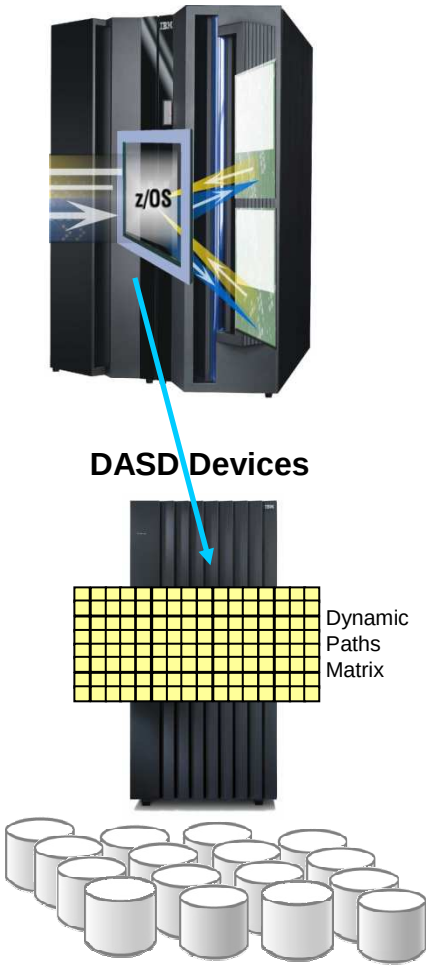
- The process of determining path availability is referred to as Pathing
  - during this process MVS will check all paths for devices genned to come up *online* by attempting to complete an I/O operation down each path defined to a device
  - if at least one path is operational the device will be online
  - Tapes are an exception: pathing is performed to offline tape devises

MVS does not report any paths or devices that are found to be not operational during pathing



DASD Pathing

- A NIP console is required before DASD pathing takes place to allow the operator to respond to out-of-line conditions encountered during the DASD pathing
  - Issue SSCH to multiple devices (test multiple devices in parallel)
  - After each successful I/O another device is tested
  - Redrive another device if an I/O is complete for a device
    - *If an I/O fails to complete within 15 seconds, the I/O operation is purged*
  - Perform path testing on each path
    - *no 1.5 sec. timeout (no IOS120A message during path testing)*
    - *create PIM (Path Installed Mask), represents CHPID's defined in IOCDS*
    - *create LPM (Logical Path Mask), used by UCB to control paths to be used for an I/O operation*
  - Get device characteristics – one path
  - Self description – each path (msg IOS291I)
  - VOLSER checking – one path for SDP products (all paths for other DASD)
    - *duplicate VOLSER message (IEA213A – not SYSRES, IEA214A SYSRES)*
  - at end of pathing wait 15 seconds for any outstanding I/O to complete
    - *mark any UCB with outstanding I/O to test later again*
    - *purge all outstanding requests*



DASD Pathing…

1 (P)	2 (D)	3 (S)	4 (V)
Path Testing	UCB Device Characteristics Initialization	SDP	VOLSER
Each Path	One Path	Each Path	One Path SDP Device Each Paths non-SDP device
CCW: 94 Release	CCWs: E4 Sense Id; 64 RDC; 54 Subsystem; FA RCD	CCWs: E4 Sense Id (one Path) FA RCD (each Path)	
Messages: IGGN504A; IGGN505A; Required Dataset missing; IOS120A moved to MSI	Message: IEC334I (duplicate SSID)	Message: IOS291I (Configuration Mismatch)	Messages: IEA213A; IEA214A (Duplicate VOLSER)

DASD pathing consists of 4 different phases: path testing on each path (P), read device characteristics (D), self-describing product (S) and VOLSER processing (V)

Any error consitions detected during the DASD pathing steps are reported to the NIP console via messages IGGN504A, IGGN505A, IEC334I, IOS291I, IEA213A or IEA214A (*any A or action messages requires operator response*)

CCW = Channel Command Word
RCD = Read Configuration Data
RDC = Read Device Characteristics
SDP = Self-describing Product
SSID = Subsystem ID (DASD CUs)

DASD Pathing…

- Dynamic Pathing Error Messages

```
IOS291 CONFIGURATION DATA COULD NOT BE READ ON PATH (24C0,49) RC=21
```

- IOS291I messages with a RC of 21, 27 or 29 indicate a possible configuration error and should be investigated

```
IEC334I DUPLICATE SUBSYSTEM X‘0001‘,CCA X‘00‘, DEVIVE 24C0 NOT BROUGHT ONLINE
```

- In addition the the IOS291I messages, a misconfiguration problem to a DASD CU may also show up as a duplicate SSID condition

```
IEA213A DUPLICATE VOLUME ‘SPOOL1‘ FOUND ON DEVICES 2465 AND 28A0
IEA213A REPLY DEVICE NUMBER WHICH IS TO REMAIN OFFLINE

IEA214A DUPLICATE SYSRES ‘S15R21‘ FOUND ON DEVICE 22C4
IEA214A VERIFY THAT THE CORRECT DEVICE WAS USED FOR IPL
IEA214A DUPLICATE DEVICE WILL REMAIN OFFLINE
IEA214A REPLY ‘CONT‘ TO CONTINUE IPL
```

- The last step of dynamic pathing is Direct Access Volume Verification (DAVV)
- DAVV processing reads the volume label of each online DASD device and updates the UCB with the VOLSER
- If a duplicate VOLSER exists, either message IEA213A or IEA214A will be issued

DASD Pathing…

- Dynamic Pathing Error Messages…

```
IGGN505A SPECIFY UNIT FOR SYS1.PRODXY.LINKLIB ON DCSYS2 OR CANCEL

R 00,1A60
IEE600I REPLY TO 00 IS;1A60
```

- If the busy condition still exists IOS120A will be issued

```
*IOS120A DEVICE 1A60 SHARED, REPLY 'CONT' OR 'WAIT'
 IOS600I REPLY TO 00 IS 'WAIT'
*IOS124A STILL WAITING FOR RESPONSE FROM DEVICE 1A60, TOTAL WAIT TIME
 IS 46 SECONDS, REPLY 'CONT' OR 'WAIT'

*IOS120A DEVICE 1A60 SHARED, REPLY 'CONT' OR 'WAIT'
 IOS600I REPLY TO 00 IS 'WAIT'
 IGGN306I 1A60,UNIT UNACCEPTABLE, 00000004
 IGGN505A SPECIFY UNIT FOR SYS1.PRODXY.LINKLIB ON DCSYS2 OR CANCEL
```

- IGGN504A or IGGN505A message issued if required dataset is on a volume that was busy during DASD pathing and the dataset is required for the IPL to complete
- Issue D U,VOL=vvvvvv on an active system that shares the DASD device to obtain the device number associated with the VOLSER

NIP RIM Processing

1. Create RTM recovery and control blocks
2. Create WTO control blocks and pools
   - WTOs issued now will be logged in SYSLOG
3. Initialize Machine Check handling (MCH)
4. Device mapping (UCWs to UCBs), test availability, and initialize non-DASD devices
5. Select and initialize NIP
   - WTOs will now be displayed on the NIP console
6. Test availability, and initialize DASD devices (DASD Pathing)
   - Operator can be prompted during validation
7. Open the master catalog
8. Create the system symbolics from IEASYMxx
9. Open SVCLIB, PARMLIB, and LOGREC
10. If required, prompt for system parameters (message IEA101A)
11. Merge and analyze the system parameters

NIP RIM Processing…

12. Initialize ASM, opening page and swap datasets
13. Process SQA= parameter
    - On a quickstart (CLPA not specified), PLPA boundaries control SQA/ESQA boundaries
    - On a coldstart, expand initial SQA/ESQA
14. Create user SVC table entries from IEASVCxx
15. Create the PLPA if CLPA specified
    - LPALSTxx datasets
    - UIM specified device support from SYS1.NUCLEUS
16. Create FLPA and MLPA, fix FLPA area and protect both areas as requested
17. Complete type 3 and 4 SVC table entries
18. Process CSA= parameter
19. Initialize system resource manager (SRM)
20. Enable RTM for task termination / SRB purge
    - Limited Function Address spaces can now be created by master scheduler
21. Initialize Cross-memory services, creates PCAUTH address space

NIP RIM Processing…

22. Initialize RSM Dataspace services, creates RASP
23. Initialize System Trace services, creates TRACE
24. Initialize Timing services, sets TOD if needed
25. Initialize SVC dump services, creates DUMPSRV address space
26. Initialize XCF/XES services, creates XCFAS address space
27. Initialize GRS services, creates GRS address space
28. Initialize SMS and PDSE services, creates SMXC and SYSBMAS address spaces
29. Open LNKLST -- drops SYS1.NUCLEUS
30. Initialize Console services, creates CONSOLE address space
    - Full function console is still unavailable
31. Initialize WLM services, creates WLM address space
32. Initialize data management
33. Initialize Concurrent-copy, creates ANTMAIN and ANTAS000 address spaces
34. Initialize UNIX System Services, creates OMVS address space

NIP RIM Processing…

35. *Close master catalog*
36. *Initialize Catalog services, creates CATALOG address space*
    - *Limited function, for use until MSI completes*
37. *Exit NIP processing*
    - *Create the IPL parameter area (IPA)*
    - *Free control blocks no longer needed by NIP*
    - *Reset traps for unexpected errors, enables full RTM recovery/retry*
    - *LINK to Master Scheduler processing*

Note: this is just a brief overview of the NIP RIMs. For a complete list of all NIP RIMs refer to the IPCS IPL statistics report at the end of this presentation

- **Master Scheduler Initialization (MSI) Overview**
  - *Completes initialization of system functions*
  - *Coordinates final completion with primary subsystem (JES2/JES3)*
- **Basic Processing**
  - *Initialize Master Trace processing*
  - *Enable full function Console processing*
    - *All MCS consoles are now available*
  - *Initialize Sysplex-wide ENF services, creates IEFSCHAS address space*
  - *Initialize MSTR subsystem*
  - *Initialize Common JES services, creates JESXCF address space*
  - *Initialize Allocation services, creates ALLOCAS address space*
  - *Attach Initiator to start Master JCL*

MSI Processing Details

1. *Initialize MIH services*
2. *Complete ASM initialization*
3. *Initialize IOS dynamic pathing, create IOSAS*
4. *Initialize Master's security environment*
5. *Initialize Console attributes, DEL=RD etc.*
6. *Initialize APPC services*
7. *Initialize TSO services*
8. *Initialize LOGREC Logstream recording*
9. *Enable ENF services*
10. *Initialize System Logger services, creates IXGLOG address space*
11. *Vary all available CPs online*
    - *we are now multiprocessing*
12. *Initialize SMF services, creates SMF address space*

MSI Processing Details…

13. Issue commands in IEACMD00 and COMMNDxx parmlib members
    - only commands processed by CONSOLE will execute now
14. Initialize RTM services
    - LOGREC recording
    - Address space termination
    - SVC dump processing
15. Initialize System security processing
16. Build defined subsystems
    - Invoke initialization routine
    - Issue START for primary JES subsystem, if requested
17. Hold primary JES STC and TSO processing
18. Indicate MSI is complete
19. Initialize Master command processing
    - Any pending commands that execute in Master will now be executed
    - Start commands are executed by Master

MSI Processing Details…

Full function address spaces can be created - JES and other tasks started under MSTR will now start

20. *Issue command processing available message*
21. *Allow pending address space creates (not done by Master) to complete*
    - *Create full function CATALOG*
    - *Original CATALOG terminates*
    - *Address spaces may switchover from limited to full function*
22. *Wait for JES to indicate primary services are available*
    - *Release primary JES STC and TSO processing*
    - *Start the System Log Syslog/OPERLOG*

All IPL processing is now complete

The next and final step is to bring up and initialize the job entry subsystem (JES2 or JES3)

IPCS Display IPL Statistic

VERBX BLSAIPST

```
*** IPL Statistics ***

IEAIPL10  00:00:00.000  ISNIRIM - Read SCPINFO
IEAIPL20  00:00:00.000  Test Block storage to 2G
IEAIPL11  00:00:00.008  Fast FIND service
IEAIPL31  00:00:00.001  LOAD service
IEAIPL30  00:00:00.001  IPLWTO service
IEAIPL46  00:00:00.110  Read SCHIBs into IPL workspace
IEAIPL49  00:00:00.000  Process Load and Default parameters
IEAIPL50  00:00:08.747  IPL parmlib - process LOADxx and NUCLSTxx
IEAIPL51  00:00:00.000  System architecture
IEAIPL43  00:00:00.012  Find and Open IODF data set
IEAIPL60  00:00:00.000  Read NCRs from IODF
IEAIPL70  00:00:00.097  UIM environment - load CBD and IOS services
IEAIPL71  00:00:00.064  Build DFT for each device
IEAIPL08  00:00:00.004  Read EDT information from IODF
IEAIPL40  00:00:00.043  Read MLTs from nucleus
IEAIPL42  00:00:00.005  Read NMLs from nucleus (IEANynnn modules)
IEAIPL41  00:00:00.627  Read PDS directory entries and CESD records
IEAIPL05  00:00:00.000  Build and sort NUCMAP
IEAIPL02  00:00:02.130  Load nucleus modules
IEAIPL04  00:00:00.005  Allocate PFT and SQA/ESQA
IEAIPL14  00:00:00.000  Build LSQA/ELSQA for Master
IEAIPL09  00:00:00.040  IAXMI - PFT, master RAB,  etc.
IEAIPL07  00:00:00.005  Update AMODE for nucleus resident SVCs
IEAIPL03  00:00:00.029  Build UCBs, ULUT, etc.
IEAIPL18  00:00:00.015  Copy and relocate EDT to ESQA
IEAIPL99  00:00:00.194  Page frame table and cleanup

Total IPL Time:  00:00:12.147
```

To determine the time required for an IPL in your installation, use

IPCS VERBX BLSAIPST

to display IPL statistics information

Note: the IPL statistic shown on this and the following slides is from a z/OS 1.10 (HBB7750) system.

IPCS Display IPL Statistic…

```
*** NIP Statistics ***

IEAVNIP0  00:00:00.034  NIP Base
IEAVNIPM  00:00:00.109  Invoke NIP RIMs
IEAVNPE6  00:00:00.065  Service Processor Interface
IEAVNPFF  00:00:00.044  Loadwait/Restart
IEAVNPA6  00:00:00.011  RTM - RTCT and recording buffer
IEAVNPC6  00:00:00.011  WTO
IEAVNPC3  00:00:00.011  Issue messages from IPL message queue
IEAVNP24  00:00:00.032  SMS Open/Mount
IEAVNP06  00:00:00.013  Machine Check
IEAVNP27  00:00:00.016  Reconfiguration
IEAVNPA2  00:01:30.319  IOS - Non-DASD UCBs
IEAVNPCA  00:00:00.219  NIP Console
IEAVNPB2  00:00:03.136  IOS - DASD UCBs
IEAVNP11  00:00:00.043  Locate and Open master calalog
IEAVNPC7  00:00:00.030  Open SYS1.SVCLIB
IEAVNPOP  00:00:00.156  Open PARMLIB
IEAVNPIL  00:00:00.091  Process IEALSTxx
IEAVNPC4  00:00:00.044  Prompt for System Parameters
IEAVNP03  00:00:00.005  Merge and analyze system parameters
IEAVNPCF  00:00:04.098  Process system name and system variables
IEAVNP76  00:00:00.057  Open LOGREC
IEAVNPE8  00:00:00.033  RSM - Process REAL=
IEAVNP23  00:00:00.050  Build GRS blocks in SQA
IEAVNP04  00:00:00.097  ASM - Open page and swap data sets
IEAVNPA8  00:00:00.012  VSM - Expand SQA
IEAVNP14  00:00:00.206  ASM part 2 - Build SQA control blocks
IEAVNPGD  00:00:00.009  Move console data to ESQA
IEAVNP25  00:00:00.618  Process SVC=
IEAVNP05  00:00:13.072  LPA, APF
IEAVNP44  00:00:00.003  ASA Reuse stuff
IEAVNPB1  00:00:00.001  Process CSCBLOC=
```

IPCS Display IPL Statistic…

```
IEAVNPE2  00:00:00.004  RACF SAF
IEAVNPB8  00:00:00.021  Create CSA
IEAVNP47  00:00:00.003  ENF
IEAVNPD6  00:00:00.002  RTM - SDUMP, ABDUMP, ESTAE
IEAVNP09  00:00:00.003  Build ASVT
IEAVNPD8  00:00:09.865  RSM - Frame queues, VRREGN= and RSU=
IEAVNP10  00:00:07.029  SRM - OPT=, IPS=, etc.
IEAVNPD1  00:00:00.022  ABDUMP
IEAVNPD2  00:00:00.025  SDUMP
IEAVNPCX  00:00:00.002  Context services, registration services
IEAVNPX1  00:00:00.002  NIP cleanup
IEAVNPF5  00:00:00.061  PCAUTH
IEAVNPF8  00:00:00.017  RASP
IEAVNP1F  00:00:00.138  SRM - I/O measurement blocks
IEAVNPC2  00:00:00.038  IOS - Move CDT to SQA
IEAVNP51  00:00:00.033  TRACE
IEAVNP20  00:00:00.231  Process CLOCK=
IEAVNP21  00:00:00.202  TOD clock
IEAVNP57  00:00:00.014  SDUMP
IEAVNPF9  00:00:15.811  XCF
IEAVNP33  00:00:13.329  GRS
IEAVNPED  00:00:00.021  PROD
IEAVNP26  00:00:01.757  SMS
IEAVNPE5  00:00:04.480  LNKLST
IEAVNPD5  00:00:00.378  Load pageable device support modules
IEAVNP88  00:00:00.238  Allocation move EDT II
IEAVNPA1  00:00:38.746  CONSOLE
IEAVNPDC  00:00:00.589  WLM
IEAVNP16  00:00:03.508  EXCP appendages
IEAVNP13  00:00:00.030  Prepare NIP/MSI interface
IEAVNP17  00:00:00.003  GTF Monitor Call interface
IEAVNPG8  00:00:00.005  VSM defined monitor call enablement
IEAVNP18  00:00:05.463  PARMLIB Scan Routine interface
```

IPCS Display IPL Statistic…

```
IEAVNPF2  00:00:00.130  Process IOS=
IEAVNP15  00:00:00.424  Process VATLST
IEAVNPRR  00:00:00.002  RRS
IEAVNPOE  00:00:00.468  USS
IEAVNPSC  00:00:00.002
IEAVNPLE  00:00:07.342  System LE RIM
IEAVNPUN  00:00:00.142  Unicode
IEAVNPXL  00:00:00.014
IEAVNP1B  00:00:00.085  Close catalog
IEAVNIPX  00:00:00.001  Nip final cleanup

Total NIP Time:  00:03:43.361
```

IPCS Display IPL Statistic…

```
*** IEEVIPL Statistics ***

IEETRACE  00:00:00.004  Master trace
ISNMSI    00:00:00.776  SPI
UCMPECBM  00:00:01.163  CONSOLE address space
ENFPC005  00:00:00.000  CONSOLE ready ENF
IEFSCHIN  00:00:00.174  IEFSCHAS address space
IEFJSINT  00:00:00.003  Subsystem interface
IEFSJLOD  00:00:00.023  JESCT
IAZINIT   00:00:00.051  JESXCF address space
IAZFSII   00:00:00.006  FSI trace
IEFQBINT  00:00:00.020  SWA manager
IEFAB4I0  00:00:00.108  ALLOCAS address space

IEEVIPL   00:00:02.332      Uncaptured time:  00:00:00.000

*** IEEMB860 Statistics ***

ILRTMRLG  00:00:00.204  ASM
IECVIOSI  00:00:42.407  IOS dynamic pathing
ATBINSYS  00:00:00.010  APPC
IKJEFXSR  00:00:00.160  TSO
IXGBLF00  00:00:00.025  Logger
COMMNDXX  00:00:00.133  COMMANDxx processing
SMFWAIT   00:00:02.534  SMF
SECPROD   00:00:04.873  Security server
IEFJSIN2  00:00:28.051  SSN= subsystem
IEFHB4I2  00:00:00.020  ALLOCAS - UCB scan
CSRINIT   00:00:00.006  Windowing services
FINSHMSI  00:00:00.000  Wait for attached CMDs

IEEMB860  00:01:18.585      Uncaptured time:  00:00:00.156

Total Time:  00:05:16.426
```

Tip: in the IPCS dialog, to display the last IPL statistic using in-storage source rather than an SVC dump, proceed as follows:

1. Select IPCS option 6 (commands)
2. Issue DROPD MAIN
3. (delete data from a previous IPCS session using in-storage as source)
4. Issue VERBX BLSAIPST MAIN

How to automatically create an IPL Statistics Report

- Wouldn't it be nice to automatically create a regular IPL statistics report with the durations of each RIM (Resource Initialization Module) during an IPL of a z/OS system?
- On the UNIX System Services Tools and Toys web site, 2 utilities are provided which can be used to create such a report
- On this web site, link IPLSTATS provides a ZIP-file, iplstats.zip, which contains these 2 utilities and a readme file:
  - IPLSTATX.OBJ
  - IPLSTATZ.OBJ
  - README
- IPLSTATX.OBJ writes the IPL statistics report to a SYSOUT dataset pointed to by the OUTPUT DD-statement
- IPLSTATZ.OBJ writes the IPL statistics report to SYSLOG/OPERLOG using WTO messages

How to automatically create an IPL Statistics Report

- **Sample report created with the IPLSTATZ utility:**

```
IPLST000I z/OS              01.11.00 CH01PROD 281700017E16   25 CPs
IPLST001I IPL started at:  2011/03/26 19:31:41.810
IPLST100I **** IPL Statistics ****
IPLST101I IEAIPL99      0.131   Page frame table and cleanup
IPLST102I               0.131   TOTAL IPL TIME (seconds)
IPLST002I NIP started at:  2011/03/26 19:32:03.588
IPLST200I **** NIP Statistics ****
IPLST201I IEAVNIP0      0.031   NIP Base
IPLST201I IEAVNIPM      0.090   Invoke NIP RIMs
IPLST201I IEAVNPE6      0.077   Service Processor Interface
IPLST201I IEAVNPFF      0.031   Loadwait/Restart
IPLST201I IEAVNPA6      0.007   RTM - RTCT and recording buffer
IPLST201I IEAVNPC6      0.014   WTO
IPLST201I IEAVNPC3      0.007   Issue messages from IPL message queue (IEA371I)
IPLST201I IEAVNP24      0.040   SMS Open/Mount
IPLST201I IEAVNP06      0.009   Machine Check
IPLST201I IEAVNP27      0.013   Reconfiguration
IPLST201I IEAVNPA2     52.271   IOS - Non-DASD UCBs
IPLST201I IEAVNPCA      0.007   NIP Console
IPLST201I IEAVNPB2      1.794   IOS - DASD UCBs
IPLST201I IEAVNP11      0.457   Locate and Open master catalog
```

How to automatically create an IPL Statistics Report

- **Sample report created with the IPLSTATZ utility (cont.):**

```
IEAVNPC7       0.049   Open SYS1.SVCLIB
IEAVNPOP       0.145   Open PARMLIB
IEAVNPIL       0.802   Process IEALSTxx
IEAVNPC4       0.156   Prompt for System Parameters
IEAVNP03       0.029   Merge and analyze system parameters
IEAVNPCF      33.371   Process system name and system variables
IEAVNP76       0.160   Open LOGREC
IEAVNPE8       0.014   RSM - Process REAL=
IEAVNP23       0.045   Build GRS blocks in SQA
IEAVNP04       0.058   ASM - Open page and swap data sets
IEAVNPA8       0.008   VSM - Expand SQA
IEAVNP14       1.138   ASM part 2 - Build SQA control blocks
IEAVNPGD       0.003   Move console data to ESQA
IEAVNP25       5.454   Process SVC=
IEAVNP05      21.837   LPA, APF
IEAVNP44       0.013   ASA Reuse stuff
IEAVNPB1       0.002   Process CSCBLOC=
IEAVNPE2       0.004   RACF SAF
IEAVNPB8       0.020   Create CSA
IEAVNP47       0.007   ENF
IEAVNPD6       0.002   RTM - SDUMP, ABDUMP, ESTAE
```

How to automatically create an IPL Statistics Report

- **Sample report created with the IPLSTATZ utility (cont.):**

```
IEAVNP09      0.003   Build ASVT
IEAVNPD8      3.656   RSM - Frame queues, VRREGN= and RSU=
IEAVNP10     33.031   SRM - OPT=, IPS=, etc.
IEAVNPD1      0.032   ABDUMP
IEAVNPD2      0.019   SDUMP
IEAVNPCX      0.002   Context services, registration services
IEAVNPX1      0.002   NIP cleanup
IEAVNPF5      0.051   PCAUTH
IEAVNPF8      0.032   RASP
IEAVNP1F      0.076   SRM - I/O measurement blocks
IEAVNPC2      0.017   IOS - Move CDT to SQA
IEAVNP51      0.038   TRACE
IEAVNP20      2.198   Process CLOCK=
IEAVNP21      0.484   TOD clock
IEAVNP57      0.014   SDUMP
IEAVNPF9     31.598   XCF
IEAVNP33     12.009   GRS
IEAVNPED      0.027   PROD
IEAVNP26      7.928   SMS
IEAVNPE5      4.943   LNKLST
IEAVNPD5      0.470   Load pageable device support modules
```

How to automatically create an IPL Statistics Report

- **Sample report created with the IPLSTATZ utility (cont.):**

```
IEAVNP09      0.003   Build ASVT
IEAVNP88      0.160   Allocation move EDT II
IEAVNPA1    274.898   CONSOLE
IEAVNPDC      0.524   WLM
IEAVNP16      3.646   EXCP appendages
IEAVNP13      0.078   Prepare NIP/MSI interface
IEAVNP17      0.003   GTF Monitor Call interface
IEAVNPG8      0.005   VSM defined monitor call enablement
IEAVNP18     69.774   PARMLIB Scan Routine interface
IEAVNPF2      0.091   Process IOS=
IEAVNP15      0.815   Process VATLST
IEAVNPRR      0.002   RRS
IEAVNPOE      0.405   USS
IEAVNPSC      0.007   Metal C RTL
IEAVNPLE     59.583   System LE RIM
IEAVNPUN      0.320   Unicode
IEAVNPXL      0.053   zXML Parser
IEAVNPCI      0.002   IQP
IEAVNPDD      0.002   DDM
IEAVNP1B      0.102   Close catalog
IEAVNIPX      0.001   NIP final cleanup
            574.339   TOTAL NIP TIME (seconds)
```

How to automatically create an IPL Statistics Report

- **Sample report created with the IPLSTATZ utility (cont.):**

```
**** IEEVIPL Statistics ****
IEETRACE      0.002   Master trace
ISNMSI        0.768   SPI
UCMPECBM      0.548   CONSOLE address space
ENFPC005      0.000   CONSOLE ready ENF
IEFSCHIN      0.208   IEFSCHAS address space
IEFJSINT      0.002   Subsystem interface
IEFSJLOD      0.021   JESCT
IAZINIT       0.035   JESXCF address space
IAZFSII       0.006   FSI trace
IEFQBINT      0.024   SWA manager
IEFAB4I0      0.113   ALLOCAS address space
IEEVIPL       1.727   TOTAL TIME. Uncaptured time:      0.000
```

How to automatically create an IPL Statistics Report

- **Sample report created with the IPLSTATZ utility (cont.):**

```
IPLST003I MSI started at:  2011/03/26  7:36:50.088
IPLST400I **** IEEMB860 Statistics ****
IPLST401I ILRTMRLG      0.180   ASM
IPLST401I IECVIOSI     35.865   IOS dynamic pathing
IPLST401I ATBINSYS      0.017   APPC
IPLST401I IKJEFXSR      0.153   TSO
IPLST401I IXGBLF00      0.016   Logger
IPLST401I HWIAMIN1      0.021   BCPii
IPLST401I COMMNDXX      0.142   COMMANDxx processing
IPLST401I SMFWAIT       2.656   SMF
IPLST401I SECPROD       3.914   Security server
IPLST401I IEFJSIN2     14.911   SSN= subsystem
IPLST401I IEFHB4I2      0.009   ALLOCAS - UCB scan
IPLST401I CSRINIT       0.004   Windowing (DIV) services
IPLST401I FINSHMSI      0.000   Wait for attached CMDs
IPLST004I MSI ended at:  2011/03/26  7:37:48.109
IPLST402I IEEMB860     58.021   TOTAL TIME. Uncaptured time:      0.133
IPLST009I IPL completed (TCP/IP up) at:  2011/03/26  7:44:08.813
IPLST999I             634.217   TOTAL TIME (seconds) IPL+NIP+VIPL+MB860
IPLST999I            1028.468   ELAPSE TIME (seconds) to TCP/IP up
```

How to automatically create an IPL Statistics Report

- **Implementation:**

1. Download iplstats.zip from the USS Tools and Toys web site to your workstation:
   - http://www-03.ibm.com/systems/z/os/zos/features/unix/bpxa1ty2.html
2. Unzip iplstats.zip
3. Binary transfer IPLSTATZ.OBJ to a PDS or PDSE library with RECFM=FB and LRECL=80
4. Link-edit (bind) IPLSTATZ into a (LNKLST) load library
5. Create a started task procedure named IPLSTATZ and store it in a procedure library, e.g. SYS1.PROCLIB:

```
//IPLSTATZ PROC
//*************************************************************
//* REF: SYS1.PROCLIB(IPLSTATZ).                              *
//* DOC: CREATE AN IPL STATISTICS REPORT AND WRITE IT TO      *
//*      SYSLOG/OPERLOG USING WTO MESSAGES.                   *
//*************************************************************
//IEFPROC EXEC PGM=IPLSTATZ,REGION=0M,TIME=NOLIMIT
```

   - Note: add //STEPLIB DD DSN=… if IPLSTATZ has not been placed into a LNKLST library

6. Add the following statement to the COMMND00 parmlib member:

```
COM='S IPLSTATZ'
```

Where to read more

- z/Architecture Priciples of Operations", SA22-7832
- zEnterprise System Support Element Operations Guide, SC28-6896
- ABCs of z/OS System Programming Volume 1, SG24-6981
- z/OS Planned Outage Avoidance Checklist, SG24-7328
- System z Mean Time to Recovery Best Practices, SG24-7816

Terms and Abbreviations

- **ASM** Auixiliary Storage Manager
- **ENF** Event Notification Facility
- **IOCP** I/O Configuration Program
- **IOS** Input/Output Supervisor
- **IRIM** IPL Resource Initialization Module
- **MCH** Machine Check Handler
- **MIH** Missing Interrupt Handler
- **NIP** Nucleus Initialization Phase
- **RIM** Resource Initialization Module
- **RTM** Recovery Termination Manager
- **SRM** System Resource Manager
- **SYSRES** System residence Volume
- **UCB** Unit Control Block
- **UIM** Unit Information Module

DAT Dynamic Address Translation
IOCDS I/O Configuration Data Set
IODF I/O Definition File
IPL Initial Program Load
JES Job Entry Subsystem
MCS Multiple Console Support
MSI Master Scheduler Initialization
POR Power-on-Reset
RSM Real Storage Manager
SMS System managed Storage
SVC Supervisor Call
TOD Time of Day Clock
UCW Unit Control Word
VSM Virtual Storage Management

The End...