

Episode 8
Bit Bucket - Behind the Eight Ball

Session 4817
SHARE 83
Boston, MA
12 August, 1994

Carl Youngren
State of California
Health and Welfare Data Center
Technology Division (MS 513)
1651 Alhambra Boulevard
Sacramento, CA 95816
(916)739-7660
IBMMAIL(USHW14PC)
cyoungre@hw1.cahwnet.gov

Bob Shannon
Deluxe Corporation
1050 W. County Rd. F
St. Paul, MN 55126
(612)481-4392
IBMMAIL(USDELHBL)
usdelhbl@ibmmail.com

Trademarks

ACF/VTAM

CICS/ESA

DB2

DFSMS

EMIF

Enterprise Systems Architecture / 370

Enterprise Systems Architecture / 390

Enterprise Systems Connection
Architecture

ESA/370

ESA/390

ES/3090

ES/9000

ESCON

Hardware Configuration Definition

Hiperbatch

Hipersorting

Hiperspace

IBM (R) IMS/ESA

MVS/DFP MVS/ESA

MVS/SP Netview

OPC OS/2

PS/2 PR/SM

SAA SystemView

Omegamon, OMII MVS, OMII CICS, OMII DB2, OMII VTAM, and AF/Operator are
copyrighted by Candle Corporation

VM/ESA

ASTEX, PDSMAN, Automate, SAR and Insite/DB2 are copyrighted by Legent
Corporation

VTAM

ADABAS is copyrighted by Software AG

HSC is copyrighted by STK

SHRINK is copyrighted by Sterling Software

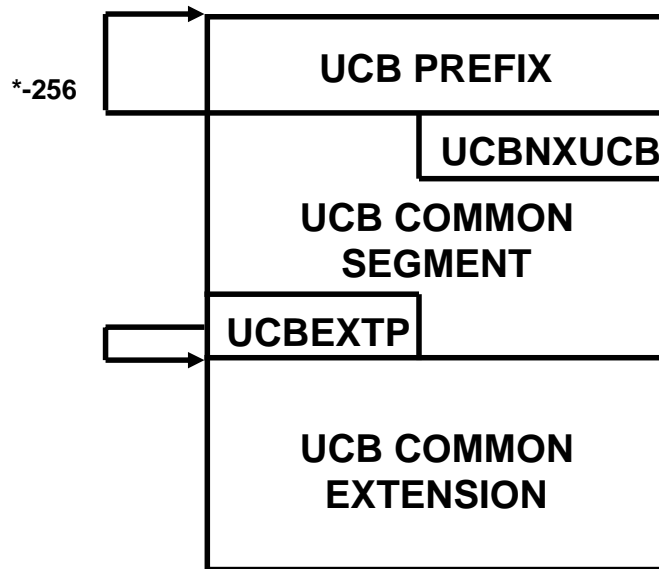
Hiperstation is copyrighted by Peregrine Systems

The Monitor is copyrighted by Landmark Corporation.

SAS/PC is copyrighted by the SAS Institute

XPAF is copyrighted by Xerox Corporation

UCB Structure in SP 4.3



UCB Prefix:

- Hot I/O indicator
- IOS IOQ chain pointer
- Subchannel data

Common Segment:

- Device Number
- UCB ID
 - X'FF' - actual UCB
 - X'CC' - copy of UCB
- Status Flags
- Device Type
- Pointer to next UCB
- Pointer to UCB Extension

UCB Extension:

- Separate mapping by device type, DASD, TAPE, etc.
- For DASD, contains:
 - Volser
 - Mount attributes
 - # open DCBs, etc.

Dynamic Reconfiguration Management

Facilitates changing the I/O configuration without POR or IPL
Device states:

- **Static:** does not support dynamic capability
- **Installation-static:** device supports dynamic capability but installation specifies non-dynamic
- **Dynamic:** device supports dynamic and installation specifies as dynamic

UCB Pinning: marks a device ineligible for deletion

Pin Token: 8 byte token from a pin operation; input to **UNPIN**

I/O Configuration Token:

- 48 byte token which reflects current I/O configuration
- initially set at IPL
- used to insure that programs manipulating devices know the current I/O configuration

UCB-Related Services

Macro	Replaces	Function
UCBSCAN	CVTUCBSC	Returns UCB Copy
UCBLOOK	IOSLOOK	Returns UCB Address
UCBDEVN	n/a	Returns EBCDIC devno
EDTINFO	IEFEB4UV	Returns EDT information
IOCINFO	n/a	Return I/O config token
UCBPIN	n/a	Prevents device deletion
CONFCHG	n/a	Rqst config chng notif

3995-153 Optical Storage

Re-writable optical storage - looks like a 3390

Fills the price/performance gap between tape and DASD

Speeds and Feeds:

- 2 pickers
- 4 read/write stations
- 3 cartridges comprise 1 logical 3390 volume
- Small footprint - 27.2"W x 37.1"D x 79.9"H
- Holds 144 cartridges - 178 GB of storage
- 113 Expansion Unit available to hold another 144 cartridges

S - L - O - W compared to DASD - impacts backup strategy:

- Back data migrated to optical while it's still on DASD
- Backup active data weekly; backup the transactions daily

Size is different than 3390s:

- 4366 cylinders per logical volume (using double capacity cartridges)
- 1.9 x 3390-2
- 1.3 x 3390-3
- Impacts offsite recovery; to insure that data can be restored to conventional 3390s, insure that no dataset is larger than a 3390-2

PSF3995 on MKTTOOLS is a SAS/Assembler package to obtain 3995 statistics:

- Queries the control unit
- Output processed by SAS code into a SAS database
- Source code *is* included
- IMHO 3995 statistics should be available via an inquiry to RMF; a non-standard tool is not satisfactory

UCB-Related Problems We've Encountered

We use the LSPACE Command Processor to obtain device space information

Problem #1: LSPACE CP could not find UCBs defined with HCD

- LSPACE followed pointers anchored off the CVT

CVTUCBA	=>	1st UCB
UCBMXUCB	=>	Next UCB
UCBEXT	=>	UCB Extension
UCBPREFIX	=	UCBOB-UCBPFLN

- IBM-provided services *must* be used for HCD-defined devices

Problem #2: Henceforth *all* pointers will be invalid:

- UCB Prefix won't be located at UCBADDR-256
- Pointers to the UCB Extension won't be valid

Problem #3: We could not distinguish a 3995-153 optical storage device from a 3390:

- The LSPACE CP issues the LSPACE macro which reads the VTOC
- When processing a list of devices, (e.g., SYSDA esoteric) including optical could cause many cartridges to be mounted, hence causing performance degradation

Problem # 4: LSPACE CP didn't work with 3390-9 DASD because the number of cylinders increased from four digits to five

UCBSCAN Service

Returns UCBs based on caller-specified criteria

- Can return either a UCB address or a UCB copy
- Can limit search to DASD, TAPE, CTC, ALL etc.
- Can limit search to static devices or can include dynamic

UCBs are returned one at a time

- Must re-invoke UCBSCAN for every UCB
- However, UCBSCAN will automatically increment to the next device number

Caller must specify beginning device number; (zero indicates start at the first device)

Caller must supply a 100 byte workarea

- *Must* be initialized to binary zeros prior to first invocation
- *Only* initialize the workarea prior to the first invocation; if you reinitialize each time you will get the first device forever

To use UCBSCAN for a single device, specify the devno; check the devno returned to insure it matches

Can be invoked in Problem or Supervisor State

UCBSCAN Coding Example

```

XC      SCAN_PARMLIST(L'SCAN_PARMLIST),SCAN_PARMLIST

LOOP   XC      UCBAREA,(L'UCBAREA),UCBAREA
        XC      RETCODE('RETCODE),RETCODE
        XC      REASON,('REASON),REASON
        XC      DEVNO,(L'DEVNO),DEVNO

UCBSCAN COPY,                                return ucb copy          X
        WORKAREA=WORKAREA,                    100 byte work area     X
        UCBAREA=UCBAREA,                       48 bytes for ucb copy  X
        DYNAMIC=YES,                            handle dynamic ucbs    X
        DEVCLASS=DASD,                          return dasd only       X
        RANGE=ALL,                               do 3 & 4 byte devnos  X
        DEVN=DEVNO,                             first device to process X
        IOCTOKEN=IOCTOKEN,                      i/o config token      X
        RETCODE=RETCODE,                        return code             X
        RSNCODE=REASON                          reason code

LTR     R15,R15                                work ok?
BNZ     ERROR_RTN                              bif no

LA      R3,UCBAREA                             get addr of ucb copy
USING  UCBCMSEG,R3                             map it

ERROR_RTN DS 0H
        CLC     RETCODE,FOUR                    all done?
        BE      FINISHED                       bif yes

SCAN_PARMLIST DS 52F                            ucbscan parmlist
        ORG SCAN_PARMLIST                      reset loc counter

WORKAREA DS CL100                               workarea
UCBAREA DS CL48                                return ucb copy here
IOCTOKEN DS CL48                               config token
DEVNO DS CL2 device number                    device number
        DS CL2
RETCODE DS CL4 return code                    return code
REASON DS CL4 reason code                    reason code

```

UCBLOOK

Returns address of UCB Common Segment

Similar to UCBSCAN with five important differences:

- Must run authorized, i.e., Supervisor State or Key 0-7
- Selection criteria must be explicitly changed to process additional UCBs on subsequent invocations
- Can specify volser as the selection criteria
- Can specify an EBCDIC device number as the selection criteria
- *Cannot* return a UCB Copy

	X	LOOK_PARMLIST(L'LOOK_PARMLIST),LOOK_PARMLIST		
LOOP	MVC	DEVNCHAR,EBCDIC_ZEROS	move zero to devnchar	
	MODESET	MODE=SUP	empower thyself	
	UCBLOOK	DEVNCHAR=DEVNCHAR,	EBCDIC devno	X
		UCBPTR=UCB_ADDRESS,	addr of ucb	X
		DYNAMIC=YES,	do dynamic ucbs	X
		RANGE=ALL,	do 3&4 digit devices	X
		NOPIN,	do not pin	X
		IOCTOKEN=IOC_TOKEN,	i/o token	X
		RETCODE=RETCODE,	return code	X
		RSNCODE=REASON	reason code	
LOOK_PARMLIST	DS	16F	ucblook parmlist	
		ORG LOOK_PARMLIST	reset loc counter	
UCB_ADDRESS	DS	F	ucb address	
IOC_TOKEN	DS	CL48	config token	
DEVNCHAR	DS	CL4	EBCDIC device number	
RETCODE	DS	CL4	return code	
REASON	DS	CL4	reason code	

UCBINFO

Five functions:

- **DEVCOUNT** - return count of devices
- **PRFXDATA** - return info from UCB Prefix (IOSDUPI)
- **PATHMAP** - return info about a path for a device (IOSDMAP)
- **PATHINFO** - superset of pathmap (IOSDPATH)
- **GETCDR** - unintended interface for CDR

Can be invoked in Problem or Supervisor State

Can hold no lock higher than IOSULUT

```

UCBINFO PRFXDATA,                get ucb prefix          x
      UCBPTR=UCB_ADDRESS,        addr of ucb            x
      UCBPAREA=UCB_PREFIX_AREA,  area for prefix        x
      RETCODE=RETURN_CODE,       return code            x
      RSNCODE=REASON_CODE        reason code
      LA      R3,UCB_PREFIX_AREA  point to prefix
      USING   IOSDUPI,R3         map it
  
```

```

UCBINFO PATHINFO,                rqst path + intrfce info x
      UCBPTR=UCB_ADDRESS,        addr of ucb            x
      PATHAREA=PATH_INFO_AREA,   target area for info   x
      RETCODE=RETURN_CODE,       return code            x
      RSNCODE=REASON_CODE        reason code
      LA      R2,PATH_INFO_AREA   map path map area
      USING   IOSDPATH,R2        map it
  
```

```

UCB_ADDRESS      DS      F      ucb address
UCB_PREFIX_AREA  DS      CL48    target for ucb prefix
PATH_INFO_AREA   DS      CL256   target for pathinfo
RETURN_CODE      DS      F      return code
REASON_CODE      DS      F      reason
IOSDPATH
IOSDUPI
  
```

If You're in a Hurry

IOSCMXA and IOSUPFA are *fastpath* services to obtain the UCB Extension and the UCB Prefix respectively:

- Intended to provide high performance
- Can be invoked in Problem or Supervisor State
- Caller must provide recovery environment
- Caller must pin the UCB or simply not care
- *Neither* can use a UCB Copy

IOSCMXA UCBPTR=UCB_ADDRESS, UCBCXPTR=EXTENSION_ADDR, RETCODE=RETURN_CODE, REASON=REASON_CODE	addr of ucb return extension addr return code reason code	X X X
---	--	-------------

IOSUPFA UCBPTR=UCB_ADDRESS, UCBPADDR=PREFIX_ADDR	addr of ucb return prefix addr	X X
---	-----------------------------------	--------

UCB_ADDRESS	DS	F	ucb address
EXTENSION_ADDR	DS	F	extension addr
PREFIX_ADDR	DS	F	prefix addr
RETURN_CODE	DS	F	return code
REASON_CODE	DS	F	reason code

EDTINFO

Provides information about the Eligible Device Table

Functions:

- **CHKGRPS** - determine if a specific device number constitutes a valid allocation group
- **CHKUNIT** - determine if a device number matches a unit name
- **RTNNAMD** - return list of generic device types or esoteric names
- **RTNUCBA** - return ucb address
- **RTNUNIT** - return unit names for a device number
- **RTNGRID** - return allocation group id
- **RTNATTR** - return general information about a unit name or device number
- **RTNDEVN** - return device numbers for unit name
- **MAXELIG** - return highest device number in an allocation group

Note: the list of devices from RTNDEVN is in device preference order, *not* devno order

In order to get devices in devno order, use UCBSCAN followed by EDTINFO w/CHKUNIT

EDTINFO Coding Example

Array Size	
# Entries	
F	Devno 1
F	Devno 2
F	Devno n

```

XC    EDT_PARMLIST(L'EDT_PARMLIST),EDT_PARMLIST
LA    R2,DEVNLIST_LEN                length for getmain
STORAGE OBTAIN,ADDR=(R1),LENGTH=(R2),COND=YES,    x
      LOC=(BELOW,ANY),BNDRY=PAGE

ST    R0,0(R1)                        store length
ST    R1,EDT_DEVNLIST                 store address
MVC   EDT_UNITNAME,ESOTERIC_NAME     indicate esoteric

EDTINFO RTNDEV,                       return device numbers
      DYNAMIC=YES,                   do dynamic devices
      UNITNAME=EDT_UNITNAME,         esoteric name
      DEVNLIST=EDT_DEVNLIST,         addr for device list
      IOCTOKEN=EDT_IOCTOKEN,        I/O config token
      RETCODE=EDT_RETCODE,          return code
      RSNCODE=EDT_REASON             reason code

LTR   R15,R15
BNZ   CHECK_EDTINFO_RC

EDT_PARMLIST DS 17F                    parmlist
      ORG EDT_PARMLIST                reset location counter
EDT_UNITNAME DS CL8                    unit name
EDT_IOCTOKEN DS CL48                  I/O config token
EDT_DEVNLIST DS F                      addr of device list
EDT_RETCODE  DS F                      return code
EDT_REASON   DS F                      reason code
DEVNLIST_LEN DC AL4(8+(4*2046))       length for getmain
ESOTERIC_NAME DC CL8'SYSOPTIC'       esoteric
  
```

DEVTYPE Macro

```
DEVTYPE ,(DEVTYPE_AREA,DEVTYPE_AREA_LEN) ,           X
        UCBLIST=(DEVTYPE_UCBADDR,1),                 X
        MF=L
```

```
DEVTYPE_UCBADDR  DS  F
DEVTYPE_AREA     DS  CL24
DEVTYPE_AREA_LEN EQU *-DEVTYPE_AREA
```

```
DEVTYPE_UCBADDR1 DS  F           ucb address
DEVTYPE_AREA     DS  CL28       target area for data
DEVTYPE_AREA_LEN EQU *-DEVTYPE_AREA target area length
```

```
*psuedo code to map DEVTYPE_AREA           *
    use IHADVA for actual mapping          *
    note: if DEVTYPE is coded as above, only 24 bytes of data *
    are returned; optional parameters will return additional data
```

Psuedo mapping - use IHADVA

```
DEVICE_DESCRIPTION DS  F           device description (UCBTYP)
MAX_BLOCK_SIZE     DS  F           maximum block size #_CYLINDERS
                   DS  H           # cylinders on device
#_TRKS_PER_CYL     DS  H           # tracks per cylinder
MAX_TRACK_LENGTH   DS  H           maximum track length
BLOCK_OVERHEAD     DS  H           block overhead
CHARACTERISTICS    DS  F           device characteristic flags
SECTOR_INFO        DS  F           sector information
                   ORG SECTOR
SECTOR_OVERHEAD    DS  H           sector overhead
#_SECTORS          DS  CL1         # sectors for the device
#_DATA_SECTORS     DS  CL1         # data sectors
```

Building a Device Table

D3995-153	DS	0F	3995-153
	DC	X'00'	filler
	DC	X'F'	device type
	DC	H'4366'	X'110E' cylinders
	DC	CL8'3995-153'	device description
*			
D3390-9	DS	0F	3390-9
	DC	X'00'	filler
	DC	X'F'	device type
	DC	H'10020'	X'2724' cylinders
	DC	CL8'3390-9'	device description
*			
D3390-3	DS	0F	3390-3
	DC	X'00'	filler
	DC	X'F'	device type
	DC	H'3340'	X'D0C' cylinders
	DC	CL8'3390-3'	device description
*			
EOT	DS	0F	end of table
	DC	X'00'	filler
	DC	X'FF'	end of table indicator
	DC	X'FFFF'	bogus cylinders
	DC	CL8'EOT'	end of table

Device Characteristics

Type	Cyls	Description	Type	Cyls	Description
X'4'	2156	9345-2	X'4'	1440	9345-1
X'F'	4366	3995-153	X'F'	10020	3390-9
X'F'	3340	3390-3	X'F'	2227	3390-2
X'F''	1113	3390	X'E'	2656	3380-8
X'E'	886	3380	X'B'	555	3350

Match devtype from the table to UCBTBYT3 and match # Cyls from table to DVACYL

LSPACE Macro

SVC 78

Used to retrieve volume freespace information

UCB must be *below* the 16Mb line

Can use a UCB Copy

Information returned either as data (binary) or as a message (EBCDIC)

LSPACE

```
,UCB={addr |(reg)}  
,MSG={{addr | (reg | 0)}  
,EXPMSG={{addr | (reg) | 0}  
,DATA={addr | (reg) | 0}  
,SMF={TEST | YES | NONE}  
,F4DSCB=({addr | (reg) | 0}  
,MF={E | L | I}
```

Data Area

LSPRETN	DS	CL1	return area status
LSPDSTST	DS	CI1	vtoc status
	DS	CL2	reserved
LSPDNEXT	DS	BL4	# free extents
LSPDTCLY	DS	BL4	total free cylinders
LSPDTTRK	DS	BL4	total free tracks
LSPDLCYL	DS	BL4	# cylinders in largest free extent
LSPDLTRK	DS	BI4	# free additional tracks in largest free extent
LSPDFR0S	DS	BL4	format 0 DSCB count
LSPDVIRS	DS	BL4	free VIR count
LSPDFRAG	DS	BL4	fragmentation index

LSPACE Coding Example

Most often the message area is used

To generate a parameter list, code MF=L with no parms

```
LSPACE1 LSPACE          MF=L                      generate parmlist
LSPACE2 LSPACE UCB=(R3),EXPMSG=LSPACE_EXPMSG_AREA,      x
          MF=(E,LSPACE1)
LSPACE_DATA_AREA      DS CL36
LSPACE_MSG_AREA       DS CL30
LSPACE_EXPMSG_AREA    DS CL40
LSPACE_FMT4_AREA      DS CL96
```

Sample Output:

MSG:

SPACE=2122,0072,0008,0800,0022

EXPMSG:

SPACE=002122,000072,000008,000800,000022

To support 3390-9: Use EXPMSG instead of MSG (*a real 10 minute no-brainer!*)

The "data" option hasn't changed for 3390s:

- Insure that the target area for # of Cyls is five bytes long
- Insure that you move five bytes (if an intermediate work area is used)

What Your Mother Never Told You About LXs

Linkage Indexes (LX) are integral to Cross Memory Services

- Introduced circa MVS 1.1
- Exploitation occurred with ESA

Program Call (PC) specifies LX and EX

LX Structure:

- 1024 LXs (0 - 3FF)
- Sub-divided into two types:
 - ◆ System LXs - available to all ASIDs
 - ◆ Non-System LXs - used by mutual consent
- Number of System LXs =
(SYSTEM_FUNCTION_TABLE_VALUE + NSYSLX)
 - ◆ System Function Table Value:
 - MVS 4.2.2 =18 | MVS 4.3 =20
 - ◆ NSYSLX is a customer- specified value in IEASYSxx
 - NSYSLX = nn (nn = 10 - 512, default is 55)
- Number of Non-System LXs = (1024 - # System LXs)

More About LXs

Non-System LXs located immediately after System LXs

Typically a 1:2 ratio of System to Non-System LX use

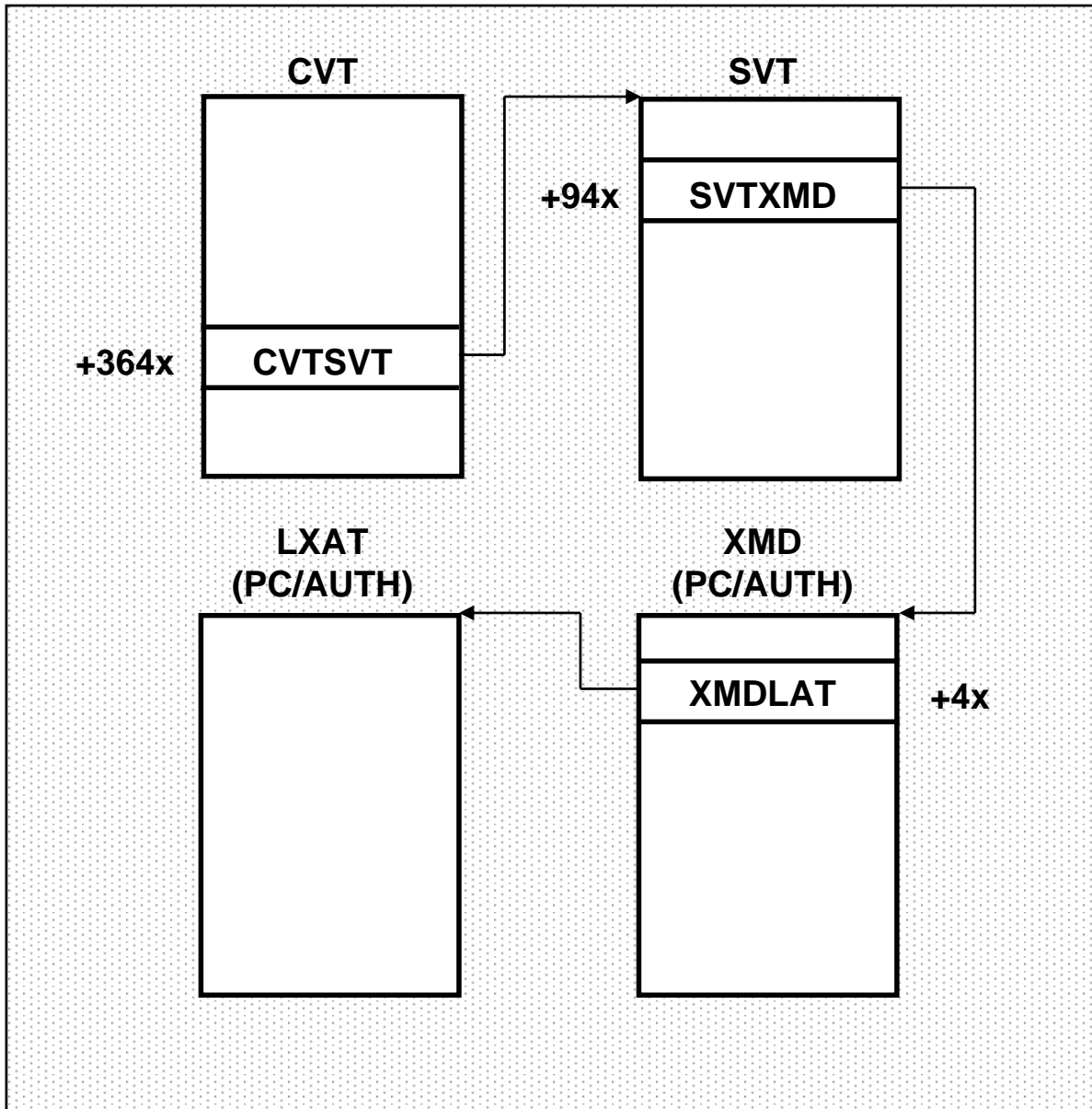
IPL required to change/reclaim LXs

- **No early warning**
- **Abend S053 RC 111/112 when LXs are exhausted**

LX use and reuse:

- **System LXs:**
 - ◆ **Cannot be reassigned**
 - ◆ **Dormant LXs can be reconnected**
 - ◆ **Remember what LX you used**
 - ◆ **Owner's ASID nonreusable**
- **Non-System LXs:**
 - ◆ **Reuseable after owner and *all* requestors have terminated**
 - ◆ **Beware of *long life* address spaces**

Control Block Structure



LXAT HDR	LXAT ACRONYM		High LX	High System LX	
	LX 0	ASID	Bind Count	ETCT	Flag
LX 1	ASID	Bind Count	ETCT	Flag	Rsvd
LX 2	ASID	Bind Count	ETCT	Flag	Rsvd
LX n	ASID	Bind Count	ETCT	Flag	Rsvd

Who Uses System LXs?

IBM Uses a bunch:

- MVS functions use approximately 26 (e.g. PCAUTH, GRS, ALLOCAS, etc.)
- APPC (2)
- TCP/IP (2)
- Catalog
- RACF
- SMS
- ANTMMAIN
- Netview (2)

ISVs:

- Candle (big time!) - OM II Subsystem, OM II/MVS/CICS/DB2/VTAM, AF/Operator
- Adabas - 1 per MPM!
- HSC - (STK Host Software Component)
- ASTEX
- Automate
- SHRINK
- PDSMAN
- SAR XMS
- Hiperstation
- Insight/DB2

Who Uses Non-System LXs?

Non-System LX Users:

- JES2 (2)
- PSF
- XPAF
- TCP/IP (3)
- SAS/PC Server
- IDMS
- CA-Dispatch
- The Monitor (3)
- OM/CICS
- OM/DB2 (1 per DB2)
- CICS
- DB2 (lots!)

Suggestions (IMHO):

- Set NSYSLX based on data
 - ◆ Reporting program available
 - ◆ Monitor use instead of waiting for ABEND053
- DB2 Users:
 - ◆ Use a batch TSO job to communicate
 - OY26621/OY64193 - SP 4.3 & DB2 2.3 are better
 - ◆ Terminate performance monitor before DB2
- Watch out for LX abusers!

An Onramp to the Information Highway

IBMMAIL now connects to the Internet

For info:

Send a note to IBMMAIL(INFORM)

No subject

/GET INET

A note with complete instructions will be sent back to you

Internet users can send to IBMMAIL users using the following address: ibmmail_id@ibmmail.com

ex: USHW14PC@IBMMAIL.COM

IBMMAIL users can register internet users by:

Sending a note to IBMMAIL(INTERNET)

/INTERNET

/REGISTER (internet id goes here)

/END

Registering an Internet user will create an IBMMAIL 8x8 id:

EX: IBMMAIL(I0123456)

Mail sent to IBMMAIL(I0123456) will be routed to the correct Internet address

Accelerating Up The Onramp

Some Internet capabilities are available without having direct Internet access

Electronic Discussion Groups (lists) are bulletin boards accessed by E-Mail

5100+ lists are available ranging from AAAE (American Association for Agricultural Education) to ZZZ-L (Zeke/Zebb/Zack Project Group)

LIST SERVERS:

Gateway to list managers (e.g., forwards subscriptions)

- May be a list manager
- Knowledgeable of lists
 - ◆ I0015251 - LISTSERV@uga.cc.uga.edu
 - ◆ I1000435 - LISTSERV@psuvm.psu.edu

Listserv Information - send "info refcard" request to a listserv

To get more information on available lists:

- List - Local lists, one line per list
- List Global - All known lists; one line per list
- List Global xyz - Only lists whose name or title contains "xyz"

Merging Into Traffic

LIST MANAGER:

One for each list

Handles administrivia:

- **SUBscribe** listname your_full_name
- **INDEX** listname
- **GET** fn ft listname
- **SIGNOFF** listname

LIST:

The bulletin board itself

To post new items send E-MAIL

New postings are distributed as E-Mail

GUIDELINES:

These are automated processes that expect simple E-Mail

- To, From, and Subject
- Anything else is a command
- *Don't* use a Prolog/Epilog
- Either upper or lower case is OK

Commands go to the List Manager id, not to the List

Send subscriptions to the List Server; save the response - it contains the E-Mail id of the manager

Pulling Into The Passing Lane

LISTS OF INTEREST:

CICS-L

RACF-L

MVS-UTIL

MVSLPD-L

TSO-REXX

DB2-L

SAG-L **(Software AG)**

ADSM-L

VSAM-L

Using XMIT/RECEIVE to Backup a PDS to a PC

1. Allocate a sequential dataset

DCB=(RECFM=FB, LRECL=80, BLKSIZE=3120)

2. Transmit a PDS to the sequential dataset

**XMIT HWDC.HWSYS1 DSNAME ('HWSYS1. JCLLIB')
OUTDSN('HWSYS1.UNLOAD.JCLLIB')**

3. Download to PC as a binary file

No ASCII conversion; no CR/LF

4. Allocate a sequential dataset on the host

DCB=(RECFM=FB, LRECL=80, BLKSIZE=3120)

5. Upload to the new sequential dataset

No ASCII conversion; no CR/LF

6. Allocate a PDS

7. Receive the dataset into the new PDS

TSO RECEIVE INDATASET ('HWSYS1. RELOAD.JCLLIB')

***INMR001I Dataset HWSYS1.JCLLIB from HWSYS1 on
SYSA***

INMR906A Enter restore parameters or 'DELETE' or 'END'

DATASET('HWSYS1.NEW.JCLLIB')

Bibliography

"DFSMS/MVS Version 1 Release 1 Using Advanced Services for Data Sets"; SC26-4921; IBM Corporation

"MVS/ESA Application Development Reference: Services for Assembler Language Programs"; GC28-1642; IBM Corporation

MVS/ESA Application Development Reference: Services for Authorized Assembler Language Programs, Volume 1"; GC28-1647; IBM Corporation

"MVS/ESA Application Development Guide: Assembler Language Programs", GC28-1644, IBM Corporation

"MVS/ESA Application Development Guide: Extended Addressability", GC28-1652, IBM Corporation

"MVS/ESA Diagnosis: System Reference", LY28-1820, IBM Corporation

"MVS/ESA System Modifications", GC28-1636, IBM Corporation

MVS/ESA: Data Areas, Volumes 1-5, LY28-1821-1825, IBM Corporation

"IBM's MVS/ESA Software Newsletter, First Quarter, 1992", IBM Corporation

"MVS ESA: Preparation for 4-Digit Device Numbers", GC28-1096, IBM Corporation

"MVS Dynamic Reconfiguration Management"; Cwiakala, Hagger, and Yudenfriend; IBM Research Journal, Volume 36 Number 4, July 1992

Channel Path Measurement Facility

The Channel Path Measurement Facility (CPMF) was introduced with EMIF

- In BASIC Mode, CPMF provides total channel busy
- In LPAR Mode, CPMF provides LPAR channel busy

Channel measurement information is contained in the Channel Path Measurement Block (CPMB) a *documented* control block introduced in MVS/SP 4.3

Requires an EMIF-capable processor

Mapped by IRACPMB

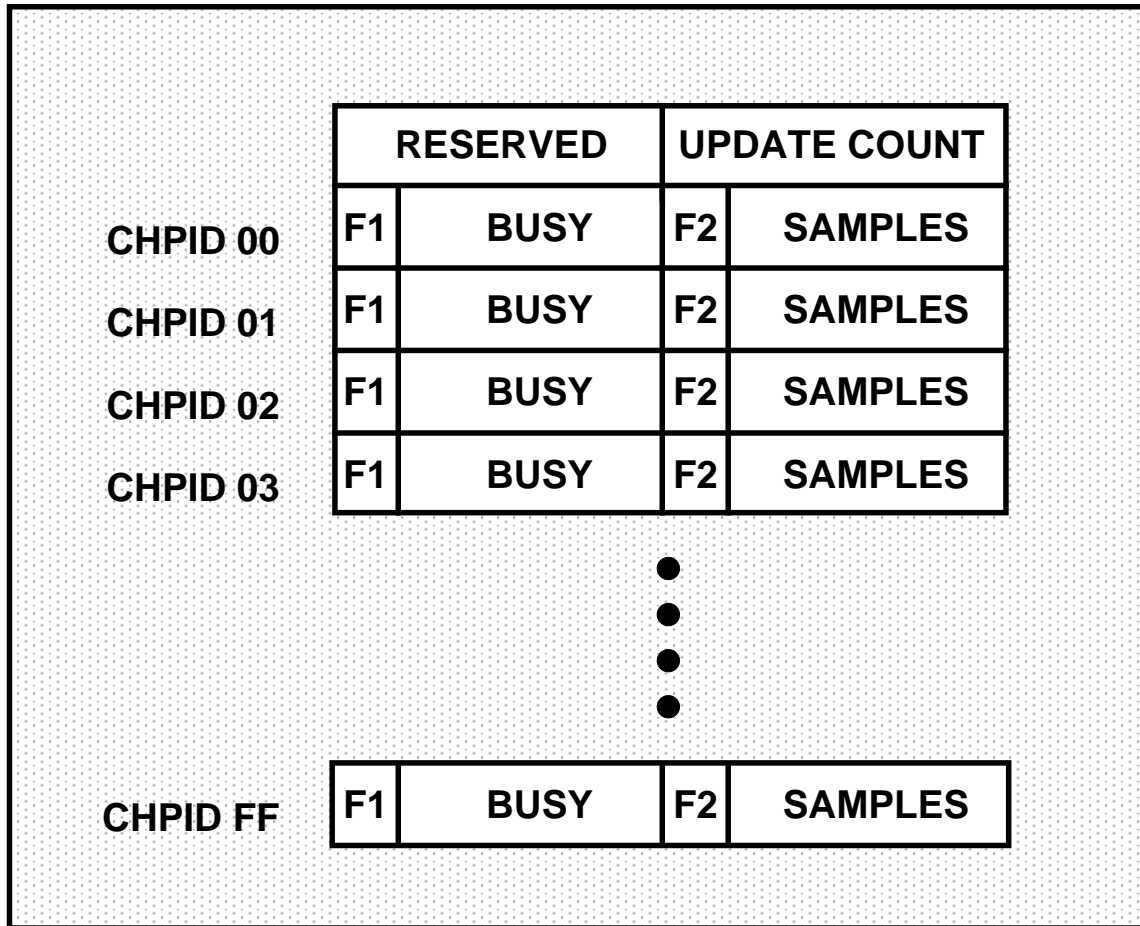
To locate the CPMB:

CVT	+	604	=	CVTOPCTP	==>	RMCT
RMCT	+	280	=	RMCTCMCT	==>	CMCT
CMCT	+	168	=	CMCTCPMB	==>	CPMB

Offsets are in decimal

Note: IRARMCT assembles RC 8

CPMB Structure



UPDATE COUNT # times CPMB updated

FLAG1: X'80' = invalid CHPID

BUSY: # times CHPID busy

SAMPLES: # times CHPID sampled

FLAG2: X'80' = shared CHPID (EMIF)

Note to MICS Users: The RMF Component must be at 9310 to support EMIF

Using the CPMB

Pseudo code for CHPID 0-255:

$\text{DELTA_BUSY} = (\text{BUSY} - \text{PREV_BUSY})$

$\text{DELTA_SAMPLES} = (\text{SAMPLES} - \text{PREV_SAMPLES})$

$\text{PCT_BUSY} = ((\text{DELTA_BUSY} / \text{DELTA_SAMPLES}) * 100)$

$\text{PREV_BUSY} = \text{BUSY}$

$\text{PREV_SAMPLES} = \text{SAMPLES}$

$\text{NEXT_CHPID} = \text{CURRENT_ADDR} + 8$

Sample LPAR Channel Busy Report

CHPID	10	S	32.0	=====>
CHPID	15	S	24.2	===>
CHPID	19	S	16.0	==>
CHPID	1C	S	27.6	=====>
CHPID	20	S	38.9	=====>
CHPID	25	S	19.7	==>
CHPID	2C	S	14.8	=>
CHPID	38	S	11.8	=>
CHPID	85		22.6	===>
CHPID	A0	S	22.8	===>
CHPID	AC	S	30.8	=====>

Sample EXEC included on Lionel's SHAREWARE Diskette