Taking the Cross out of Cross Memory
Address Spaces, Data Spaces & Access Registers (oh my!)

NaSPA May 14th, 2019

Patty Little plittle@us.ibm.com
John Shebey jshebey@us.ibm.com

Trademarks

SHARE
EDUCATE ‣ NETWORK ‣ INFLUENCE

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

- **MVS**
- **OS/390®**
- **z/Architecture®**
- **z/OS®**

* Registered trademarks of IBM Corporation

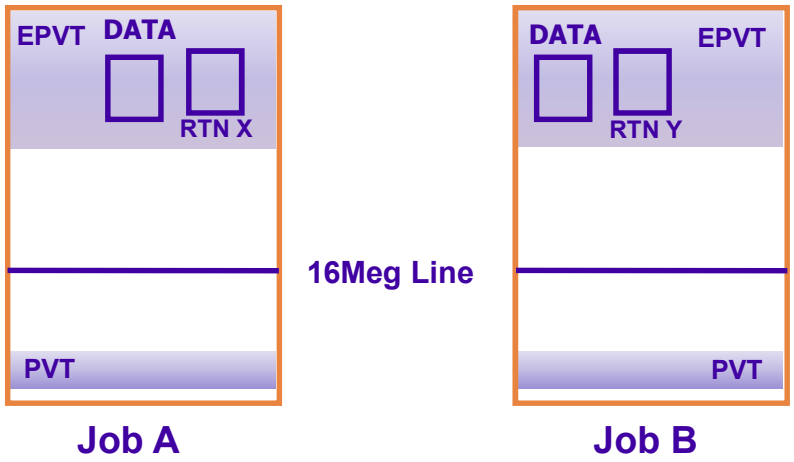
Table of Contents

Table of Contents (continued)

WHY BE CROSS?

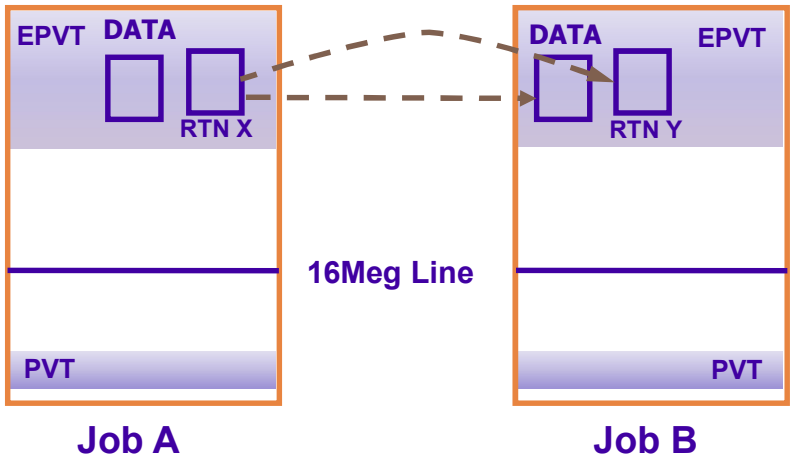
Isolation of programs and data

- Jobs/address spaces are **isolated** from each other
- Addressability to
  - Own private storage
  - Own functions/code

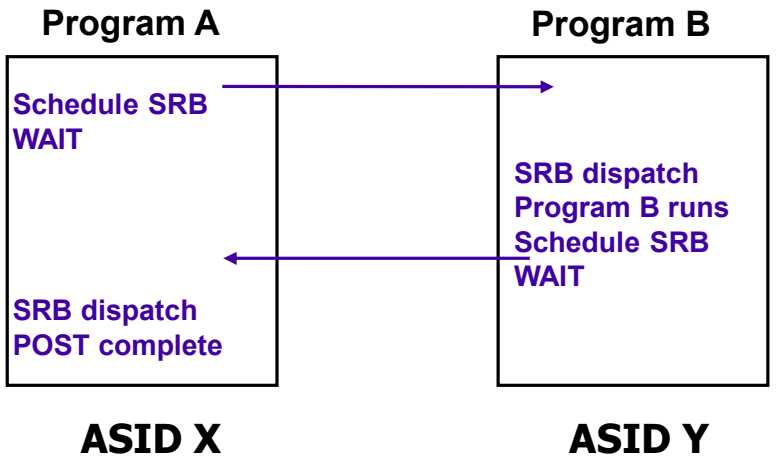


Isolation of programs and data (continued)

- Jobs/address spaces are **isolated** from each other
- Addressability to
  - Own private storage
  - Own functions/code
- Sometimes address spaces **need to interface**
  - Share data
  - Share a service or function



Asynchronous address space communication



Program A schedules an SRB which must be dispatched in the target ASID Y.

The SRB executes Program B, which schedules an SRB to ASID X to notify Program A of B's completion.

Any data shared between A and B must be in Common Storage.

Problems:

- Slow asynchronous communications
- Shared data in Common Storage exposed to other programs
- Common Storage constrained

CROSS MEMORY

What is "Cross Memory" ?

SHARE
EDUCATE ‣ NETWORK ‣ INFLUENCE

Cross memory is the mechanism to provide SYNCHRONOUS communication between address spaces for the purpose of sharing function or data.

- A program can acquire access to other address spaces' storage or code
- The program controls the cross memory environment
- Hardware/Architecture.....
  - Bits in the PSW – ASC mode
  - Control registers
  - Special instructions – PC, PR, PT, SSAR, SAC, MVCP, MVCS, etc
  - Architected structures created by z/OS, referenced by hardware
    - PC linkage tables, linkage stacks, ASTEs, authorization tables, etc

MVS Extended Addressability Guide provides details and examples for setting up a cross memory environment

Cross memory concepts

SHARE
EDUCATE ‣ NETWORK ‣ INFLUENCE

- HOME
  - the dispatched address space
    i.e. where a program starts off life, where its associated TCB resides
- PRIMARY
  - the address space where a program is presently executing
    i.e. where instructions are being fetched *
- SECONDARY
  - the address space that was primary before the last PC (program call) instruction (modifiable)

ALL programs begin life with HOME=PRIMARY=SECONDARY

***except in PSW ASC mode HOME**

How do home, primary, and secondary change?

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

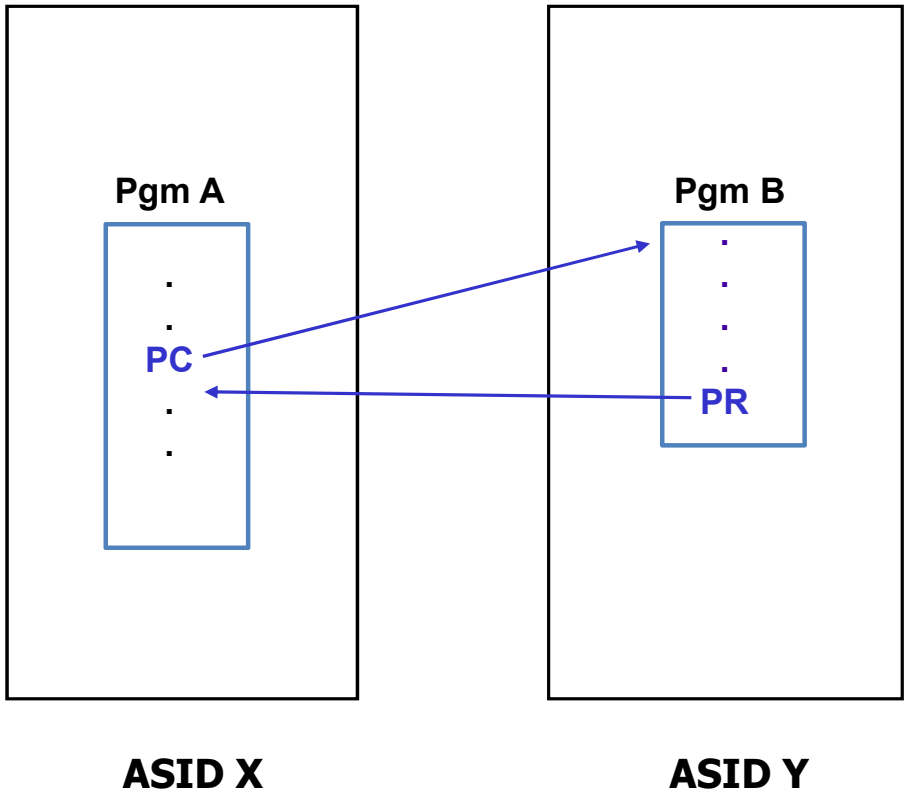
- HOME *never* changes!!
- Instructions that update PRIMARY & SECONDARY
  - PC (program call)
  - PR (program return)
  - PT (program transfer)
- The SSAR instruction updates SECONDARY

A simple example

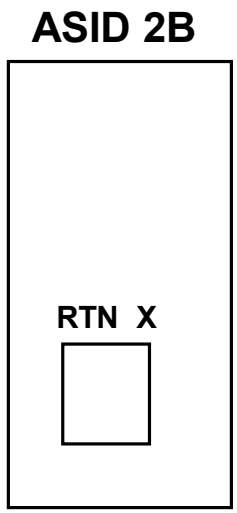
- Pgm A begins life with H=P=S=ASID X
- Pgm A issues a PC

- The PC causes a synchronous “jump” to Pgm B in ASID Y
  - Execution ASID becomes new PRIMARY
    - PRIMARY is now ASID Y
  - Previous PRIMARY becomes new SECONDARY
    - SECONDARY is now ASID X

- Pgm B completes and issues a PR
  - PRIMARY restored to time of PC
  - SECONDARY restored to time of PC
  - Processing continues at instruction after PC

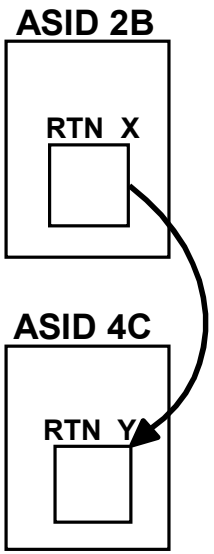


Cross memory example



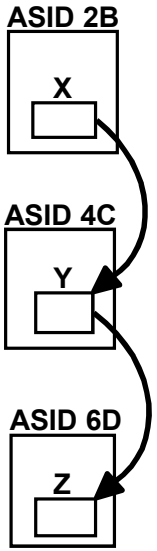
Routine X is originally dispatched

H=P=S=ASID 2B



Routine X PC's to routine Y in ASID 4C

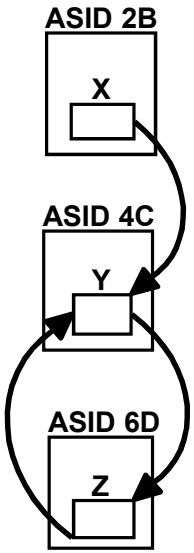
H=S=2B; P=4C



Routine Y PC's to routine Z in ASID 6D

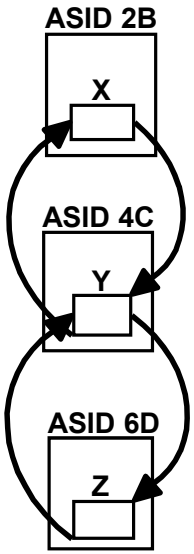
H=2B; P=6D; S=4C

Cross memory example (continued)



Routine Z finishes and does a PR

H=S=2B; P=4C

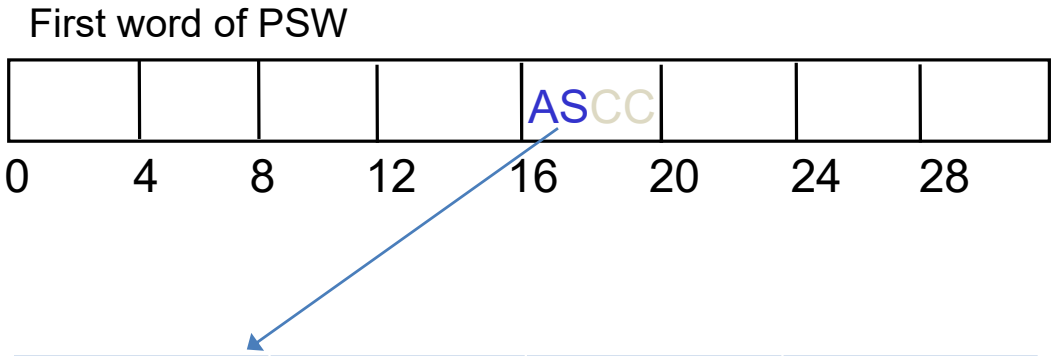


Routine Y finishes and does a PR

H=P=S=2B

Exploiting cross memory mode

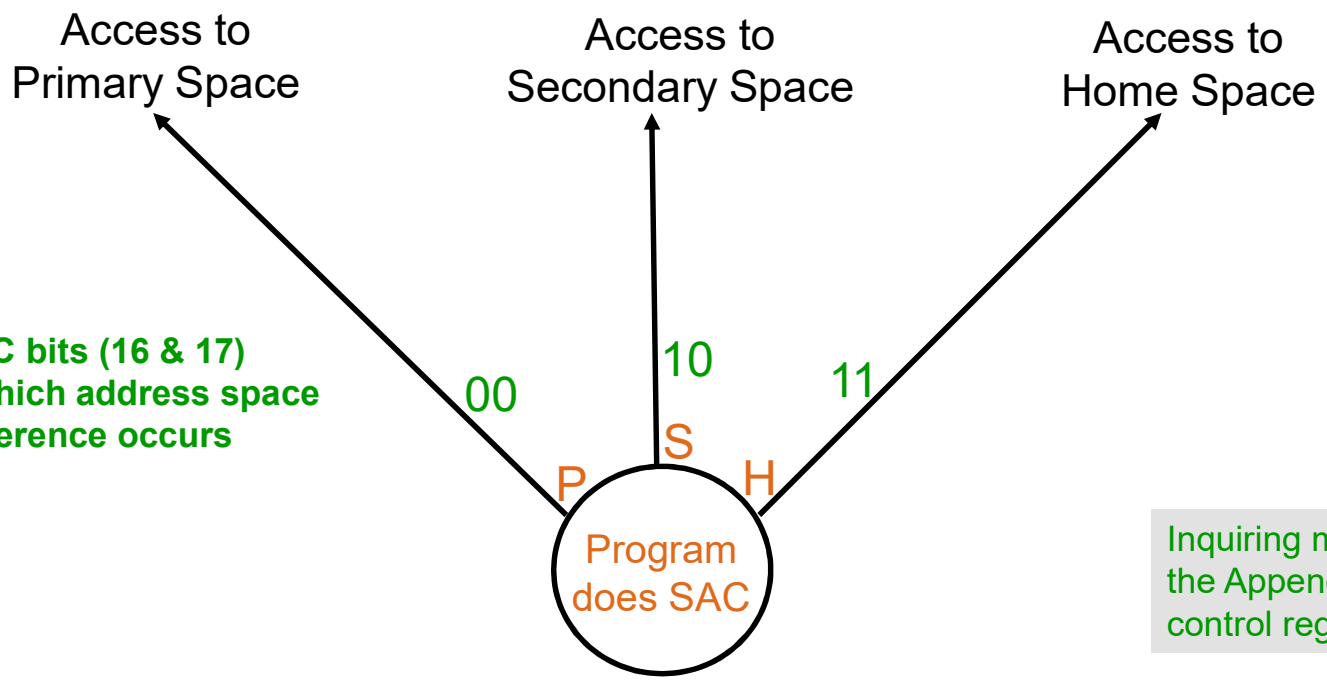
- A program can run in any one of three cross memory modes
  - Primary, Secondary or Home
- PSW Address Space Control (ASC) bits 16 & 17 indicate the cross memory mode that the program is executing in
  - Instruction fetch is always from Primary **
  - Data reference is per the program's cross memory
- Program uses the SAC instruction to switch cross memory modes



PSW ASC Bits 16&17	Cross memory mode	Instruction Fetch	Storage Access
00	Primary	Primary	Primary
10	Secondary	Primary	Secondary
11 (rare)	Home	Home **	Home

** When in Home mode (rare), instruction fetch is from the Home address space

Putting it together

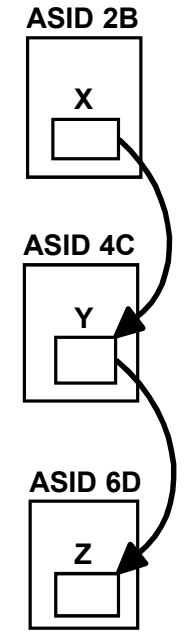


PSW ASC bits (16 & 17) indicate to which address space data reference occurs

Inquiring minds may want to refer to the Appendix for information on how control registers fit into this picture.

Instruction Fetch is always from the Primary address space! **

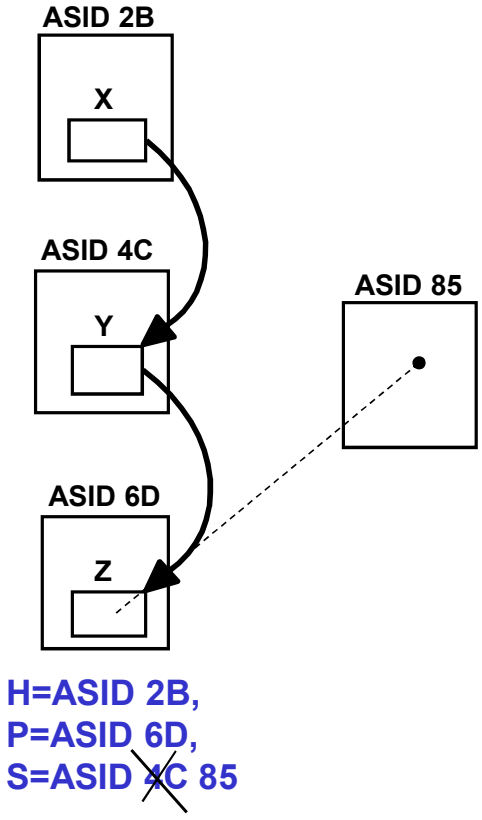
Cross memory example



H=ASID 2B
P=ASID 6D
S=ASID 4C

- Routine Z wants to move data from ASID 4C to ASID 6D
  - Use MVCP (MVC from secondary to primary)
- Routine Z wants to access data in ASID 2B
  - Issue SAC instruction to change PSW ASC bits to indicate Home mode
  - Now all data references will go to ASID 2B
- Routine Z SACs back to Primary mode

Cross memory example (continued)



- Routine Z wants to examine a bit in ASID 4C
  - Issue SAC instruction to change PSW ASC bits mode to indicate Secondary Mode
  - Issue TM
- Routine Z wants to access data in a 4th address space, ASID 85
  - Issue SSAR to 85 (assuming proper permissions)
  - Now all data references will go to ASID 85
- Routine Z SACs back to Primary mode

Spotting cross memory environment in a dump

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

Note the ASC bits 16 & 17 in PSW

```
TIME OF ERROR INFORMATION

 PSW: 040C0000 80FF4D76   INSTRUCTION LENGTH: 04   INTERRUPT CODE: 0011
 FAILING INSTRUCTION TEXT: 181458D0 F0405840 D2005820
 TRANSLATION EXCEPTION ADDRESS: 23DB5E4C

 AR/GR 0-1    00000000/00000000_A4139CBA  00000000/00000003_00075000
 AR/GR 2-3    00000000/00000000_23EC91F8  00000000/00000000_23DB5D90
 AR/GR 4-5    00000000/00000000_23DDD020  00000000/00000000_A414B85A
 AR/GR 6-7    00000000/00000000_23EC9160  00000000/00000000_00000004
 AR/GR 8-9    00000000/00000000_24139960  00000000/00000000_23EC91A8
 AR/GR 10-11  00000000/00000000_00000004  00000000/00000000_23EFE057
 AR/GR 12-13  00000000/00000000_A414AE5A  00000000/00000000_23DDE2E8
 AR/GR 14-15  00000000/00000000_00F97400  01000002/00000000_23DB5E4C

 HOME ASID: 0025     PRIMARY ASID: 0016     SECONDARY ASID: 0016
 PKM: 0000           AX: 0001               EAX: 0000
```

Note Home, Primary & Secondary ASIDs

PC routines

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

- PC instruction
  - Only operand is PC number
  - PC number identifies target routine (aka PC routine)
- Every address space has a unique PC table
  - Each entry maps a PC number to a PC routine address and ASID
    - A space switching PC "jumps" to a routine in another address space
    - A non-space switching PC "jumps" to a routine with the same address space
  - Some entries common across all address spaces, e.g. STORAGE OBTAIN = 30B
- Types of PC routines
  - Stacking - Exploits linkage stack hardware to automatically save/restore status ☺
  - Basic – Old, unfriendly, less efficient; must explicitly save/restore status ☹

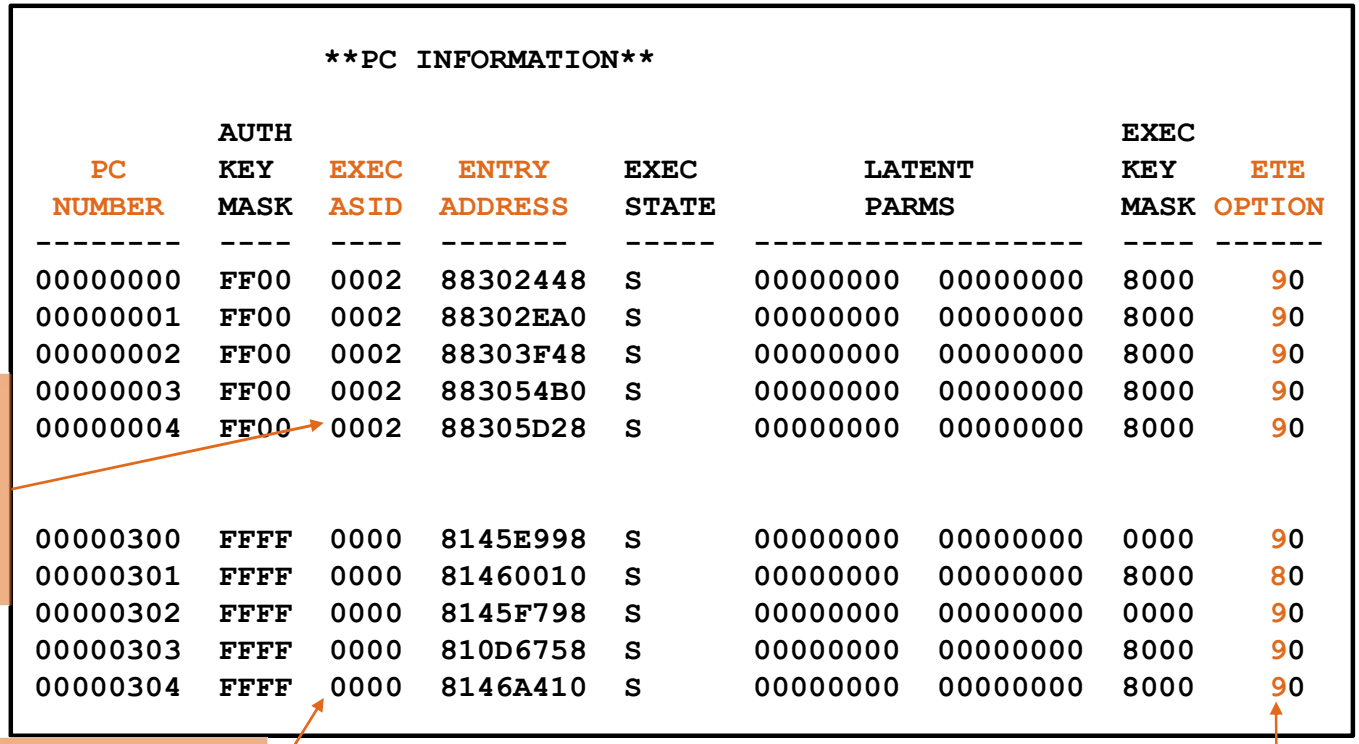
Reading a PC table

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

- To see what PC numbers are defined for a particular address space:
  - IP SUMMARY FORMAT ASID(X'yy') ←Use primary ASID
  - FIND 'PC INFORMATION' to locate the PC table

- Each entry in the PC table includes:
  - PC number
  - PC routine address (receives control when PC issued)
  - ASID that the PC routine executes in
  - Indication of stacking vs basic
  - Indication of space switching vs non-space switching
  - Information related to authorization and recovery

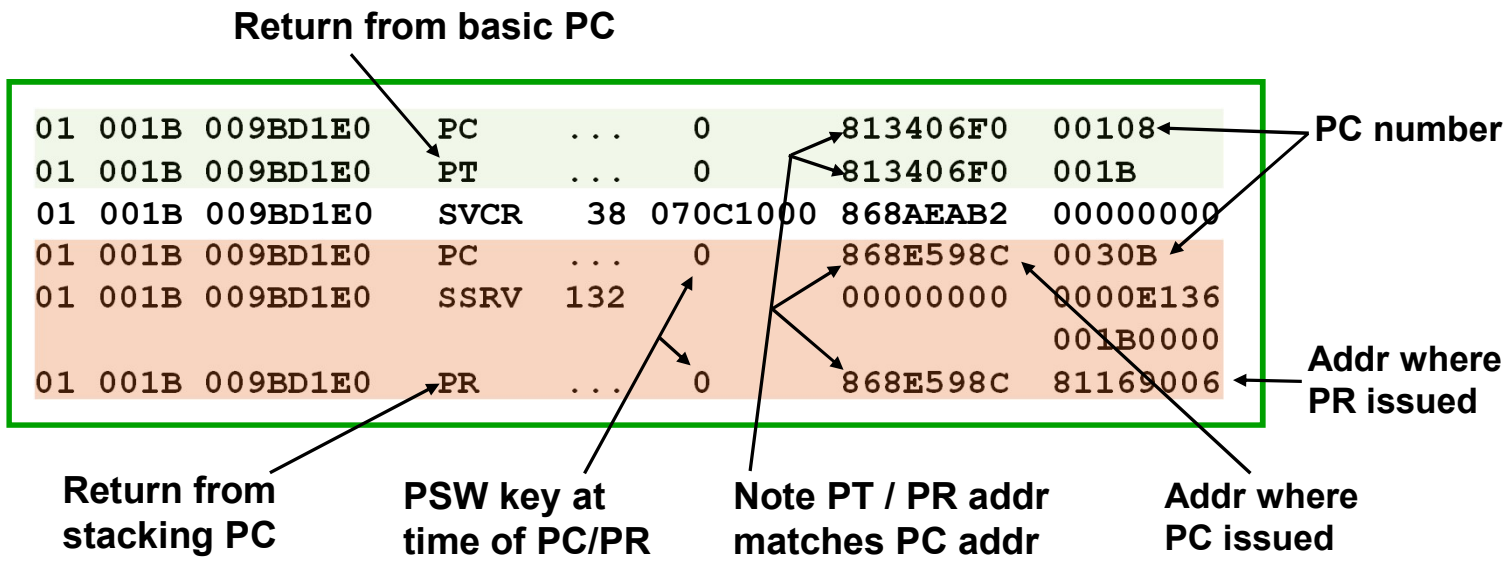
Excerpts from a PC table



```
                **PC INFORMATION**

          AUTH                                                  EXEC
   PC     KEY   EXEC   ENTRY     EXEC          LATENT         KEY    ETE
 NUMBER   MASK  ASID   ADDRESS   STATE         PARMS          MASK OPTION
--------  ----  ----  -------   -----  ----------------------  ---- ------
00000000  FF00  0002  88302448  S      00000000  00000000      8000    90
00000001  FF00  0002  88302EA0  S      00000000  00000000      8000    90
00000002  FF00  0002  88303F48  S      00000000  00000000      8000    90
00000003  FF00  0002  883054B0  S      00000000  00000000      8000    90
00000004  FF00  0002  88305D28  S      00000000  00000000      8000    90


00000300  FFFF  0000  8145E998  S      00000000  00000000      0000    90
00000301  FFFF  0000  81460010  S      00000000  00000000      8000    80
00000302  FFFF  0000  8145F798  S      00000000  00000000      0000    90
00000303  FFFF  0000  810D6758  S      00000000  00000000      8000    90
00000304  FFFF  0000  8146A410  S      00000000  00000000      8000    90
```

Reading PC information in a system trace table

PC instructions and their corresponding return instructions are traced in the system trace table:



Cross memory information in system trace

Home ASID

Primary ASID

Secondary ASID

```
PR ASID WU-ADDR- IDENT  CD/D PSW----- ADDRESS-  UNIQUE-1 UNIQUE-2 UNIQUE-3  PSACLHS- PSALOCAL PASD SASD
                                       UNIQUE-4 UNIQUE-5 UNIQUE-6  PSACLHSE

00 0010 05616F00  SRB        070C0000 814C35C0  00000010 052FA100 052FA100        00          0010 0010
                                       009D89C0 20
00 0010 05616F00  PC     ...    0      29C0F8AA              00B01           XCF Message In
00 0010 05616F00  PC     ...    0      7F697FD8              00330           TestArt  CADS=Yes
00 0010 05616F00  PR     ...    0      7F697FD8 0157A20A                                       0006
00 0010 00000000  CLKC       070C6000 FF697D0C  00001004 00FDFB70     0000  00000000 00000000 0006 0010
                                                                             00000000
00 0010 00000000  PR     ...    0      29C0F8AA 7F69848C                                       0010
```

ASID being restored as Primary

- A PC may or may not cause a space switch (check PC table to be sure!)
- Not all trace entries have a PASD/SASD ☹
- Be mindful of the PASD/SASD columns
  - Are they different from HOME?
  - If yes, this event occurred in a cross memory environment

Stacking PC's and the linkage stack

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

A linkage stack is an architected stack of save areas

- Every unit of work (TCB / SRB) has a linkage stack
- When a stacking PC is issued, the unit of work's status is automatically saved as the top entry on its linkage stack (LSE)
  - PSW
  - Registers
  - Cross memory environment
  - LSE also contains the PC number and execution ASID
- A PR restores status from that entry and removes it from the stack

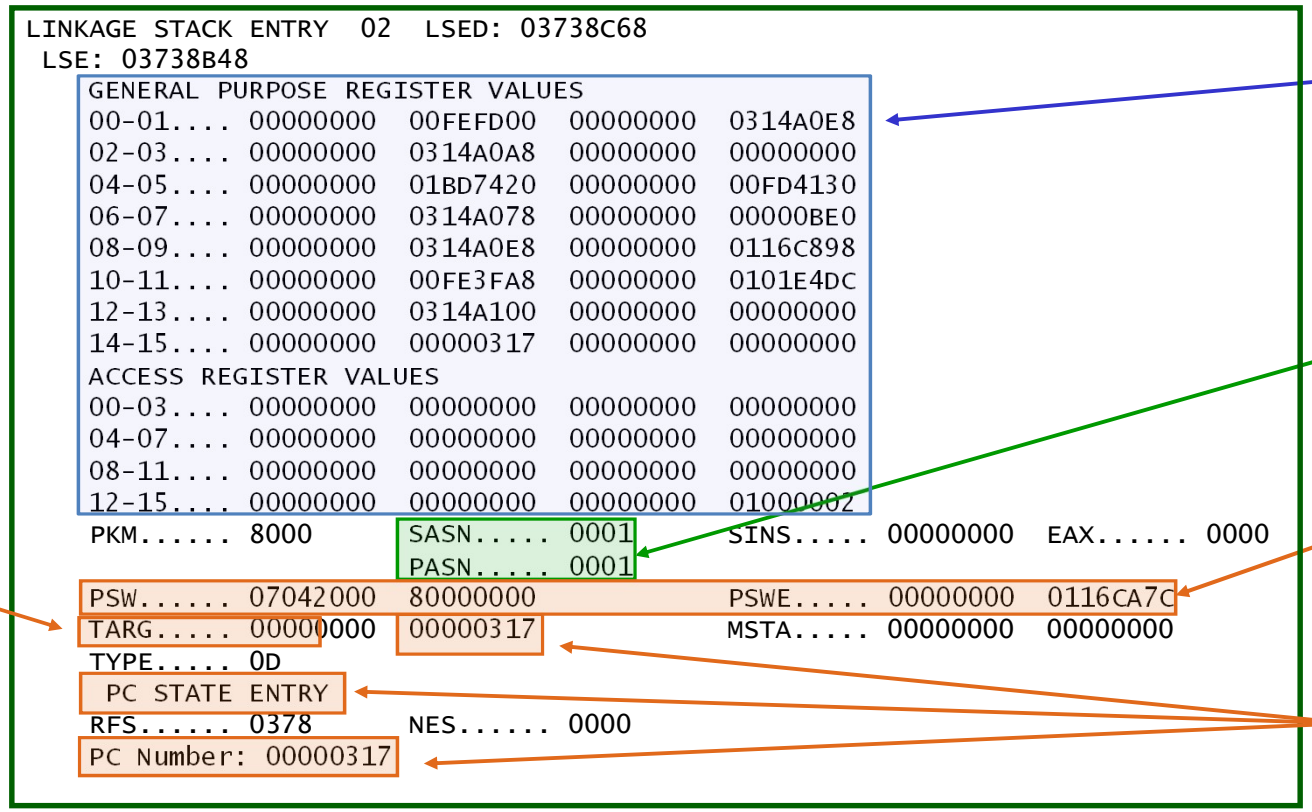
- BAKR instruction also exploits the linkage stack
  - Typically used on entry to a module in place of standard save area linkage
  - Status restored on exit from module via a PR
  - BAKR's and their corresponding PR's are not traced

Reading a TCB's linkage stack

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

- Locating a TCB's linkage stack
  - Issue: IP SUMMARY FORMAT ASID(x'yy')
  - FIND 'TCB: 00xxxxxx' where xxxxxx is the address of your target TCB
  - FIND LINKAGE to get to the linkage stack data
- Linkage stack format
  - Stack could contain many entries or none
  - Entries are formatted oldest to newest
  - Only "in use" entries are formatted

Stacking PC's and the linkage stack



Registers at time of PC

Cross memory environment at time of PC

PC is non-space switching

PSW where PC 317 was issued

PC 317

Application: a cross memory mode program check

IP STATUS FAILDATA

```
Time of Error Information

   PSW: 07041001 80000000 00000000 2108C460
   Instruction length: 06    Interrupt code: 0010
   Failing instruction text: 001CB24E 00E4D207 7010E038
   Translation exception address: 00000000_00400800

   Breaking event address: 00000000_2108B8BC
   Registers 0-7
   GR: 00000100 2300BD18 00000001 06BE56F8  00000000 00000013 2300DCF8 2680B008
   AR: 00000000 00000000 00000000 00000000  00000002 00000000 00000000 00000000
   Registers 8-15
   GR: 2300B47C 2300DF6D 2300CF6E 2300BF6F  21090248 2300AF70 004000C2 2297C110
   AR: 00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000002

   Home ASID: 00B3     Primary ASID: 0011     Secondary ASID: 0011
   PKM: 00C0           AX: 0001               EAX: 0000

   This Task's ASID/TCB: 00B3/009C5528
```

Bit 16 & 17 = 00 -> Primary mode

Program check

Failing TCB

Note cross memory environment

Application: a cross memory mode program check

IP CBF RTCT

```
RTCT: 00F55B20
   +0000  NAME..... RTCT       S
- -   - - - - - - - - - - - - - -
   +010A  ZZZ3..... 2500

          ASTB

              SDAS   SDF4   SDF5
              ----   ----   ----
          001 0011   F8     00
          002 00B3   F8     00
          003 0030   F8     00
          004 0000   00     00
          005 0000   00     00
```

What ASIDs are dumped?

IP SELECT ASID(X'11',X'B3',X'30')

```
ASID JOBNAME  ASCBADDR
---- -------- --------
0011 OMVS     00F9F200
0030 ZFS      00F88000
00B3 OPCSERV4 00F3B280
```

What are their jobnames?

Application: a cross memory mode program check

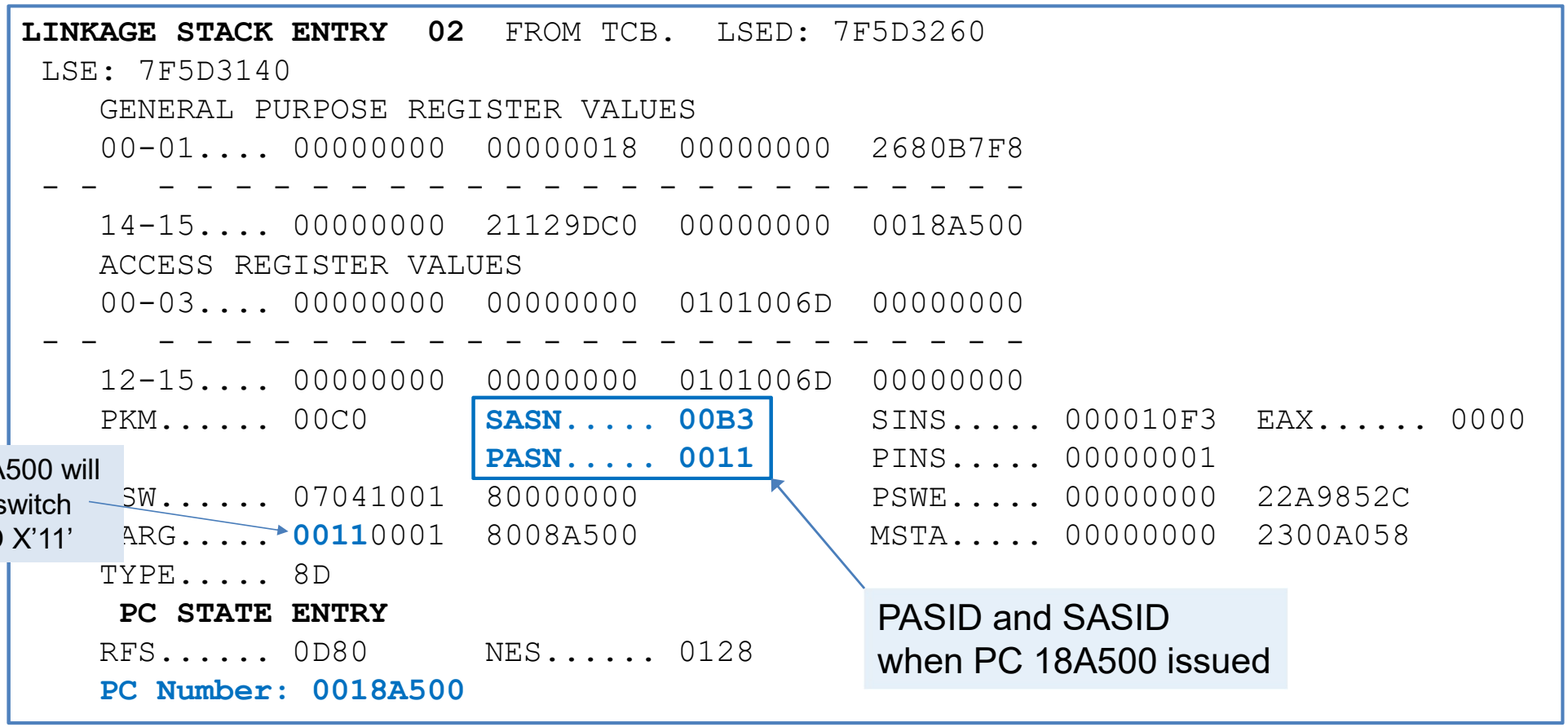
IP SUMM FORMAT ASID(X'B3'); FIND 'TCB: 009C5528'; FIND LINKAGE

```
LINKAGE STACK ENTRY  01  FROM TCB.  LSED: 7F5D3138
 LSE: 7F5D3018
    GENERAL PURPOSE REGISTER VALUES
    00-01.... 00000000  0000002B  00000000  21711A80
 - -   - - - - - - - - - - - - - - - - - - - - - - - -
    14-15.... 00000000  8A240C62  00000000  00001300
    ACCESS REGISTER VALUES
    00-03.... 00000000  00000000  00000000  00000000
 - -   - - - - - - - - - - - - - - - - - - - - - - - -
    12-15.... 00000000  00000000  00000000  00000000
    PKM...... 00C0      SASN..... 00B3      SINS..... 000010F3  EAX..... 0000
                        PASN..... 00B3      PINS..... 000010F3
    PSW...... 07851000  80000000            PSWE..... 00000000  01626686
    TARG..... 00110001  8000130B            MSTA..... 00000000  2680A058
    TYPE..... 0D
     PC STATE ENTRY
    RFS...... 0EA8      NES..... 0128
    PC Number: 0000130B
```

PC 130B will space-switch to ASID X'11'

PASID and SASID when PC 130B issued

Application: a cross memory mode program check



Application: a cross memory mode program check

```
LINKAGE STACK ENTRY  03  FROM TCB.  LSED: 7F5D3388
 LSE: 7F5D3268
    GENERAL PURPOSE REGISTER VALUES
    00-01.... 00000000  00000000  00000000  7F4054F8
 - -   - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    14-15.... 00000000  07138897  00000000  00000B80
    ACCESS REGISTER VALUES
    00-03.... 00000000  00000000  00000000  00000000
 - -   - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    12-15.... 00000000  00000000  00000000  00000000
    PKM..... 80C0       SASN..... 0011      SINS..... 00000001  EAX..... 0000
                        PASN..... 0011      PINS..... 00000001
    PSW..... 07042001   80000000            PSWE..... 00000000  07138896
    TARG.... 00000000   0713883B            MSTA..... 00000000  00000000
    TYPE.... 0C
     BAKR STATE ENTRY
    RFS..... 0C58       NES..... 0128
```

PASID and SASID when BAKR issued

Application: a cross memory mode program check

IP SYSTRACE ASID(X'B3') TCB(X'9C5528') TI(LO)

```
PR   ASID WU-Addr- Ident  CD/D PSW----- Address-  Unique-1 Unique-2 Unique-3  PSACLHS- PSALOCAL PASD SASD
0000 00B3 009C5528  PC     ...   8      01626609             01315           lseek
0000 00B3 009C5528  PR     ...   0      01626609       00_20DD601C                        00B3
0000 00B3 009C5528  PC     ...   8      01626687             0130B           read/write
0000 00B3 009C5528  PC     ...   0   3  22A9852C           0018A500
0000 00B3 009C5528 ┌PC     ...   0           00_20C3C6FC     00B33           XCF
0000 00B3 009C5528 │┌PC    ...   0           00_7F776B7A     00330           TestArt  CADS=Yes
0000 00B3 009C5528 │└PR    ...   0           00_7F776B7A 016ED824                        0006
0000 00B3 009C5528 └PR     ...   0           00_20C3C6FC 7F7734EE                        0011
0000 00B3 009C5528 ┌PC     ...   0           00_20C313B0     0030D           Wait
0000 00B3 009C5528 │SSRV   128          00000000  E34CD0D8 40000001 00000000  Wait
                   │                            00000000
0009 00B3 009C5528 │DSP        00000000_00FEC5E2  00000000 40000001 E34CD0D8  00000000 00000000 0011 0011
                   │           07041401 80000000
0009 00B3 009C5528 └PR     ...   0           00_20C313B0       00_00FEC5E8                0011
0009 00B3 009C5528  PGM    010 00000000_2108C460  00060010 00000000          00000000 00000000 0011 0011
                               07044401 80000000           00400801          00000000
0009 00B3 009C5528 *RCVY   PROG                  940C4000 00000010 00000000  00000000 00000000 0011 0011
                                                                             00000000
```

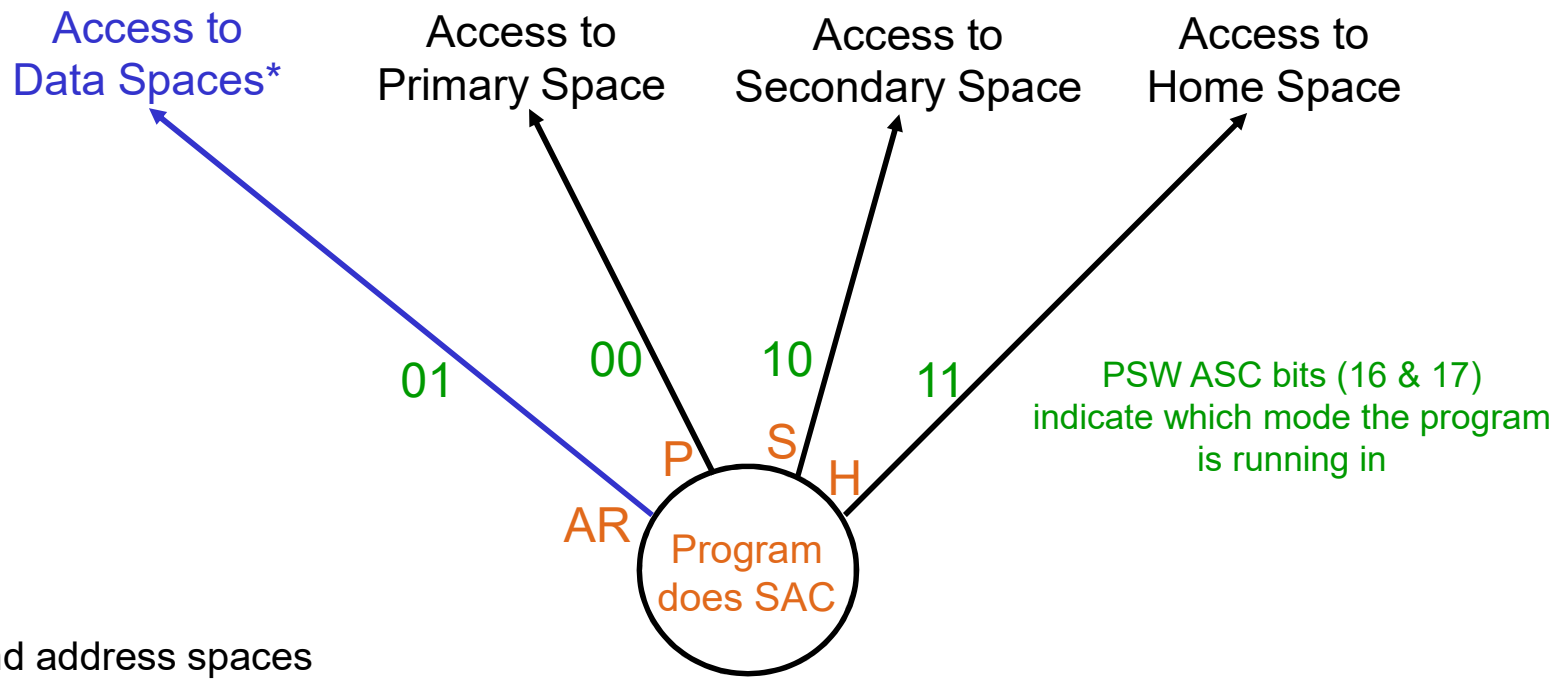
EXTENDED ADDRESSABILITY

Getting greedy:
Data spaces and access registers

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

- Problem (pre-z/Architecture): Address spaces were limited to 2Gig in size
  - Needed more private storage
  - Needed more common storage
- Solution (pre-z/Architecture): Data spaces!
  - Data-only spaces up to 2Gig in size
  - Can be used for private or common storage needs
  - Address spaces can own multiple data spaces
  - Addressable through "access register" paired with corresponding general purpose register
    - GPR identifies virtual address to access
    - AR identifies the data space that the virtual address is in
  - Only accessible by programs running in Access Register (AR) mode

Introducing Access Register (AR) mode



* And address spaces

Dramatization via chat: Setting up a data space

yo zos rsm services, i was told to talk to u about getting some storage

You've come to the right place, data spaces are us! Common or private? What size did you have in mind?

pvt, max i can get

Gotcha, 2gig it is, brb

Ok I have your data space ready. Its ID is "xyz". Now you just need to chat with Aly on the xmem service team. Give her this ID and she'll set you up with some buyer protection and give you the key to your new data space.

k ty!

Dramatization via chat: Setting up a data space

yo xmem services, was advised to chat with Aly regarding a key to my new data space

Hi, this is Aly! Welcome to xmem services. I can get you all set up. What's your data space ID?

xyz

Great! And as far as who can access your storage, which option would you like: "friends & family" or "personal use only"?

whats the dif??

Dramatization via chat: Setting up a data space

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

With “friends & family”, anyone executing in your address space can access your data space storage if you share your key with them.

With “personal use only”, only code running under your TCB can use your key.

But if you want to be able to access your data space even after you’ve PC’d into another address space, then the “personal use only” TCB option is the only way to go.

def want all my friends to be able to use this too!

Dramatization via chat: Setting up a data space

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

Ok, “friends & family” it is! One moment please while I get this set up for you.

Ok, I’ve set up your protection and activated your key. Your key code is 01xx00yy. To use it, just have your program place it in an access register, set up the corresponding general purpose register to point to the address you want to access, and use that register as the base register of an assembler instruction such as a LOAD, STORE, TM, MVC … any instruction that has a base reg!

Don’t forget you must be in AR mode for this to work!

awesome thx much!

Dramatization via chat: Setting up a data space

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

You're very welcome. Is there anything else I can help you with today?

nope ty, u've been great, working with zos service is always a fantastic experience, u guys rock!

Access lists

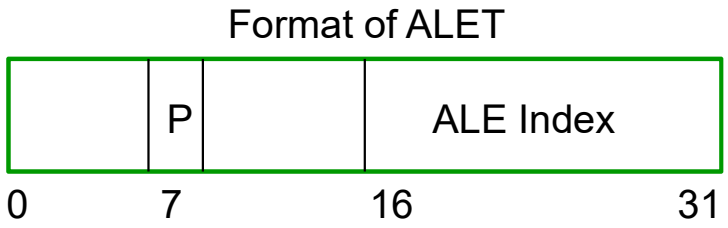
SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

- An access list is a structure that controls program access to data spaces or address spaces
  - Think of an access list as a row of gates to various destinations (data spaces or address spaces) and an ALET is a key to a gate
- Every address space has an access list
  - PASN-AL = Primary Address Space Number access list
  - Can be used by "Friends & Family", those executing in address space
- Every task and every SRB has an access list
  - DU-AL = Dispatchable Unit access list
  - Can be used only by code running under the task (TCB) or SRB

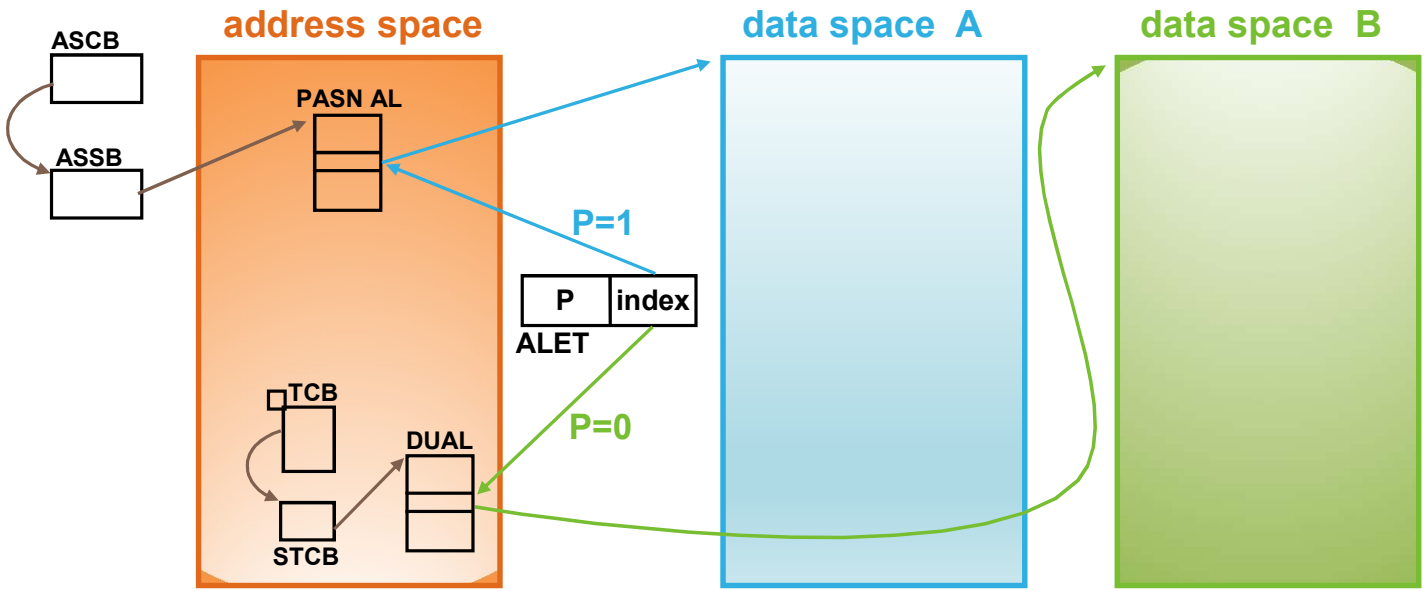
The ALET

- Pgm issues DSPSERV to define data space OR ASCRE to define addr space
  - **Output:** A unique ID called a STOKEN is returned
- Pgm issues ALESERV to set up access to data space
  - **Input:** a STOKEN and an indication of which access list to use: PASN-AL or DU-AL
  - **Output:** Indicated access list updated, ALET returned

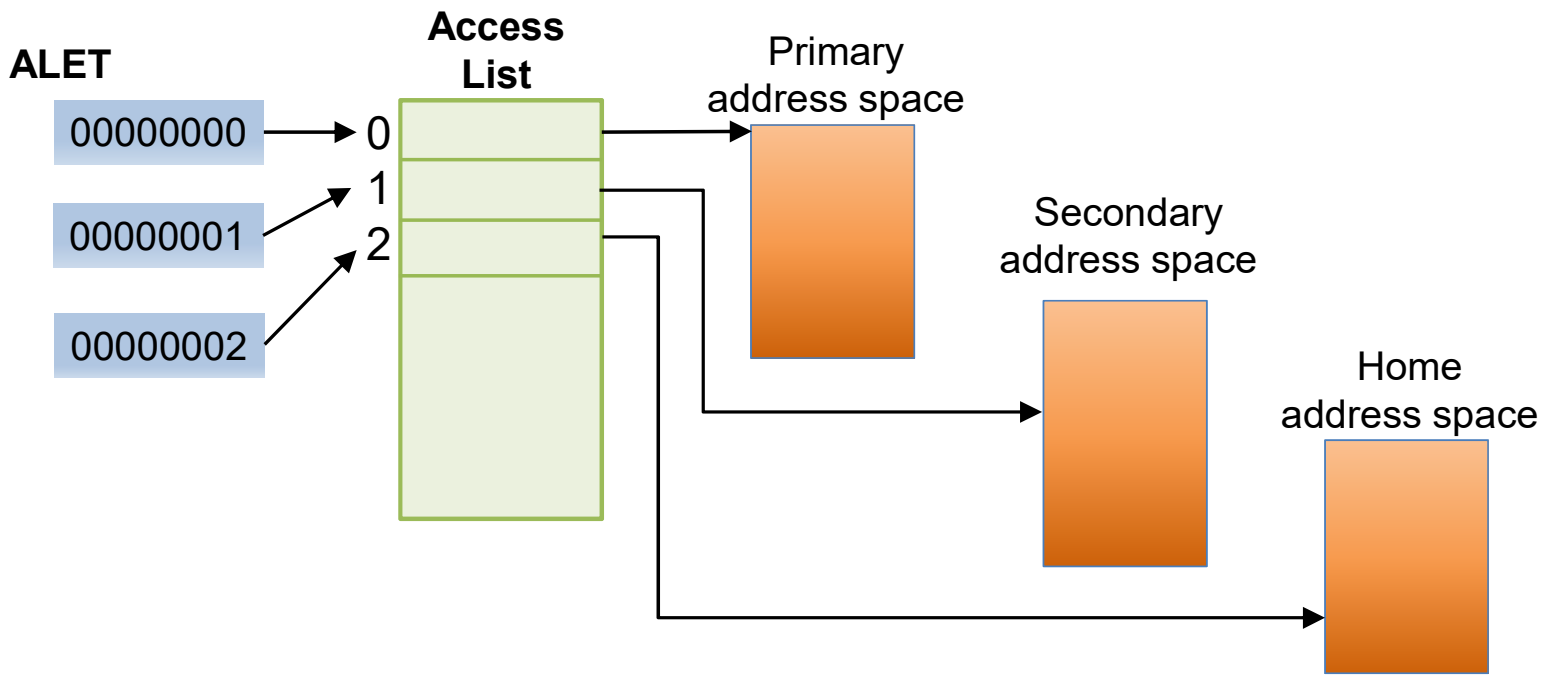
P=1 Use PASN-AL
P=0 Use DU-AL



Translating the ALET



Special ALET values



Recognizing AR mode in a dump

PSW in AR mode

Access Register

```
Time of Error Information

  PSW: 04045000 80000000 00000000 00FF1B82
  Instruction length: 04   Interrupt code: 0011
  Failing instruction text: 41F00000 185F5810 4000D203
  Translation exception address: 00000000_00000801

  Breaking event address: 00000000_00FF1A2C
  AR/GR 0-1    00000000/00000000_00F66280  00000000/00000000_0885E9A8
  AR/GR 2-3    00000000/00000000_09DCD218  00000000/00000000_80AF8310
  AR/GR 4-5    0101006B/00000000_00000008  00000000/00000000_00000000
  AR/GR 6-7    00000000/00000000_00F63E80  00000000/00000000_0257B118
  AR/GR 8-9    00000000/00000000_00FF2000  00000000/00000000_00FDDF38
  AR/GR 10-11  00000000/00000000_00000000  01FF0004/000001FF_0000023A
  AR/GR 12-13  00000000/00000000_00A9E950  00000000/00000000_0885E9A8
  AR/GR 14-15  00000000/000001F7_80FF1A1A  00000000/00000000_00000000

  Home ASID: 011F    Primary ASID: 023B    Secondary ASID: 023B
```

Finding access registers in a dump

- Access registers can be found in several status-saving control blocks
  - STCB, XSB, LSE, IHSA, and SSRB
  - SUMMARY FORMAT output includes the access registers
  - SUMMARY FORMAT will sometimes attempt access register translation
    - Following is an example from SUMM FORMAT linkage stack data:

```
PC STATE ENTRY
     RFS..... 0D80      NES..... 0000

   ACCESS REGISTER VALUES
       0-3  00000000  00000000  0101002C  00000000
       4-7  00000000  00000000  0101002D  00000000
       8-11 00000000  00000000  00000000  00000000
      12-15 00000000  00000000  00000000  00000000

 ALET TRANSLATION
 AR 02 addresses ASID(X'004F') DSPNAME(TESTSPC1)
 AR 06 addresses ASID(X'004F') DSPNAME(TESTSPC2)
```

Note that IPCS may translate ALETs that it finds in standard save areas.

Translating an ALET

Use hidden IPCS option 2.6i :

```
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      Exec      Abstract
S ALET2DSP  IAXAR2D   DataSpace Name associated with input AR/ALET
  CMD2FILE  BLSXC2FI  Writes output from an input IPCS cmd to output dataset
  CPUINFO   IEAVCPUI  displays high level CPU information
```

```
SELECT OPTION ===>

 Enter the following inputs.
 ALET Value :  ________    - ALET value should start in 01.
 ASID Value :  ____

 ALET value should be 8-digit hexadecimal (eg. 01xxxxxx).
 ASID Value should be 4-digit hexadecimal.
```

Otherwise see ARCHECK in the IPCS Commands manual

Displaying data space storage

- **To display data space storage....**
  - Issue: **IPCS LIST xxxxxxxx LEN(X'ww') ASID(X'yy') DSPNAME(zzzzzzzz)**
    - **xxxxxxxx** is the virtual storage address to be viewed
    - **ww** is the length of storage to be viewed
    - **zzzzzzzz** is the name of the data space
    - **yy** is the ASID of the address space with which the data space is associated

OR

  - Use IPCS option 1 Browse:

```
      PTR   Address              Address space
    00001 0000B000.          ASID(X'001D') DSPNAME(MYSPACE)
```

Application: an AR mode program check

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

IP STATUS FAILDATA

```
Time of Error Information

   PSW: 07044001 80000000 00000000 2108C460
   Instruction length: 06   Interrupt code: 0010
   Failing instruction text: 001CB24E 00E4D207 7010E038
   Translation exception address: 00000000_00400800
   Exception access identification: 0E

   Breaking event address: 00000000_2108B8BC
   Registers 0-7
   GR: 00000100 2300BD18 00000001 06BE56F8  00000000 00000013 2300DCF8 2680B008
   AR: 00000000 00000000 00000000 00000000  00000002 00000000 00000000 00000000
   Registers 8-15
   GR: 2300B47C 2300DF6D 2300CF6E 2300BF6F  21090248 2300AF70 004000C2 2297C110
   AR: 00000000 00000000 00000000 00000000  00000000 00000000 0101006D 00000002

   Home ASID: 00B3     Primary ASID: 0011     Secondary ASID: 0011
   PKM: 00C0           AX: 0001               EAX: 0000

   This Task's ASID/TCB: 00B3/009C5528
```

Bit 16 & 17 = 01 -> Access register (AR) mode

Program check

Access register 14 (x'E') involved in failed translation

Failing TCB

Note cross memory environment

Application: an AR mode program check

To translate a PASN-AL ALET, go to hidden IPCS option 2.6i and select ALET2DSP

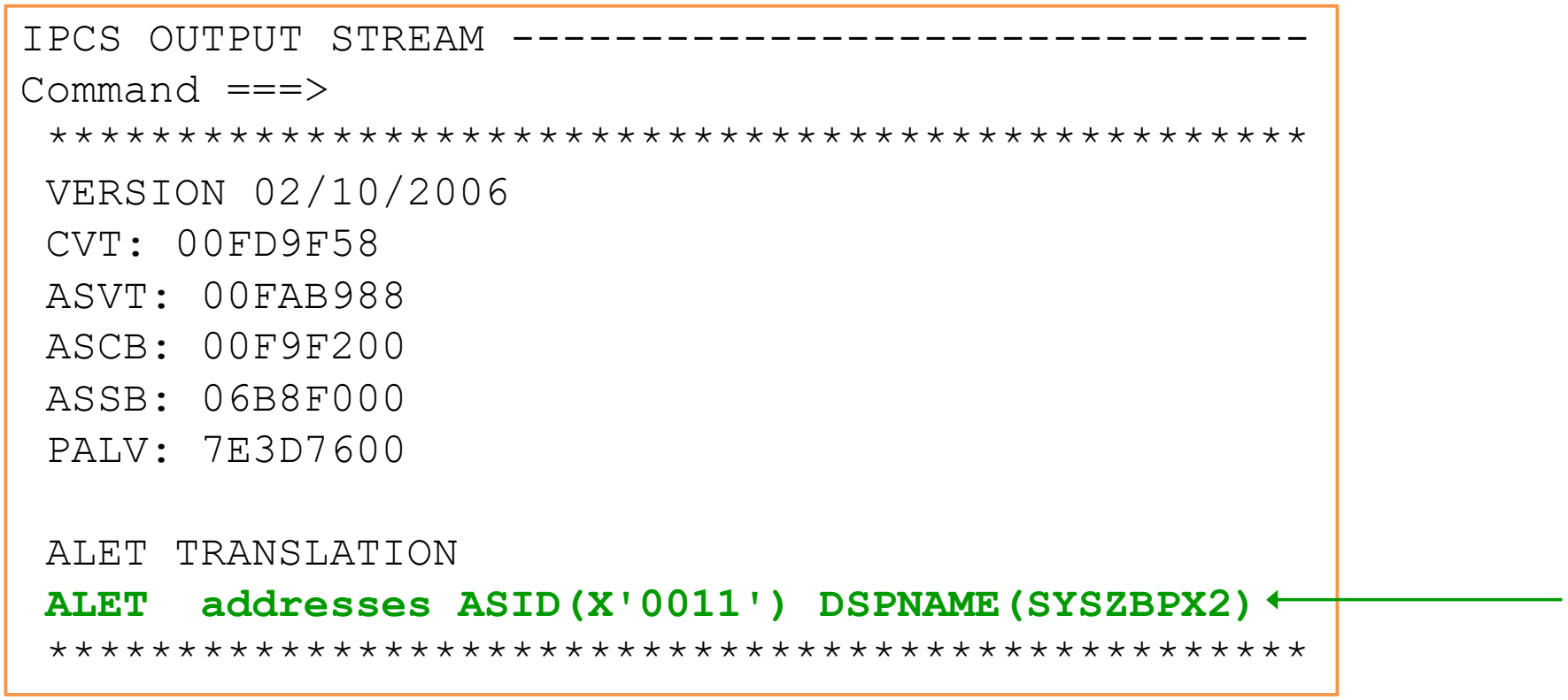
```
------------------ IPCS MVS LEVEL 2 TOOLKIT ----------------------------------
OPTION ===>                                                SCROLL ===> CSR

Level 2 toolkit functions are intended to be used as directed by service
personnel.

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      Exec      Abstract
S ALET2DSP  IAXAR2D   DataSpace Name associated with input AR/ALET
  CMD2FILE  BLSXC2FI  Writes output from an input IPCS cmd to output dataset
  CPUINFO   IEAVCPUI  displays high level CPU information
```

Select

Application: an AR mode program check

```
---------------------- IPCS - IAXAR2D EXEC -----------------------------
SELECT OPTION ===>

 Enter the following inputs.
 ALET Value :  0101006D   - ALET value should start in 01.
 ASID Value :  0011

 ALET value should be 8-digit hexadecimal (eg. 01xxxxxx).
 ASID Value should be 4-digit hexadecimal.


 IAXAR2D exec will show the output of the ARCHECK command by taking two
 input values, the ASID and ALET value, and provide the corresponding
 Dataspace name associated with that ALET value.
```

Application: an AR mode program check



Application: an AR mode program check

IPCS Browse option 1

```
Command ===>
ASID(X'0011') DSPNAME(SYSZBPX2) is the default address space
PTR   Address             Address space                              Data type
S0001 00400000.           ASID(X'0011') DSPNAME(SYSZBPX2)            AREA
      Remarks:
```

```
ASID(X'0011') DSPNAME(SYSZBPX2) ADDRESS(00.) STORAGE ----------------------
Command ===>
00400000.:0FFF. LENGTH(X'1000')--Storage not available
00401000   D6C6E2C2   00000490   00401790   FFFFFFF0   | OFSB..... .....0 |
00401010   22782CB8   22782C10   22782B68   22782AC0   | ................{ |
00401020   22782970   227828C8   22782778   227826D0   | .......H.......} |
```

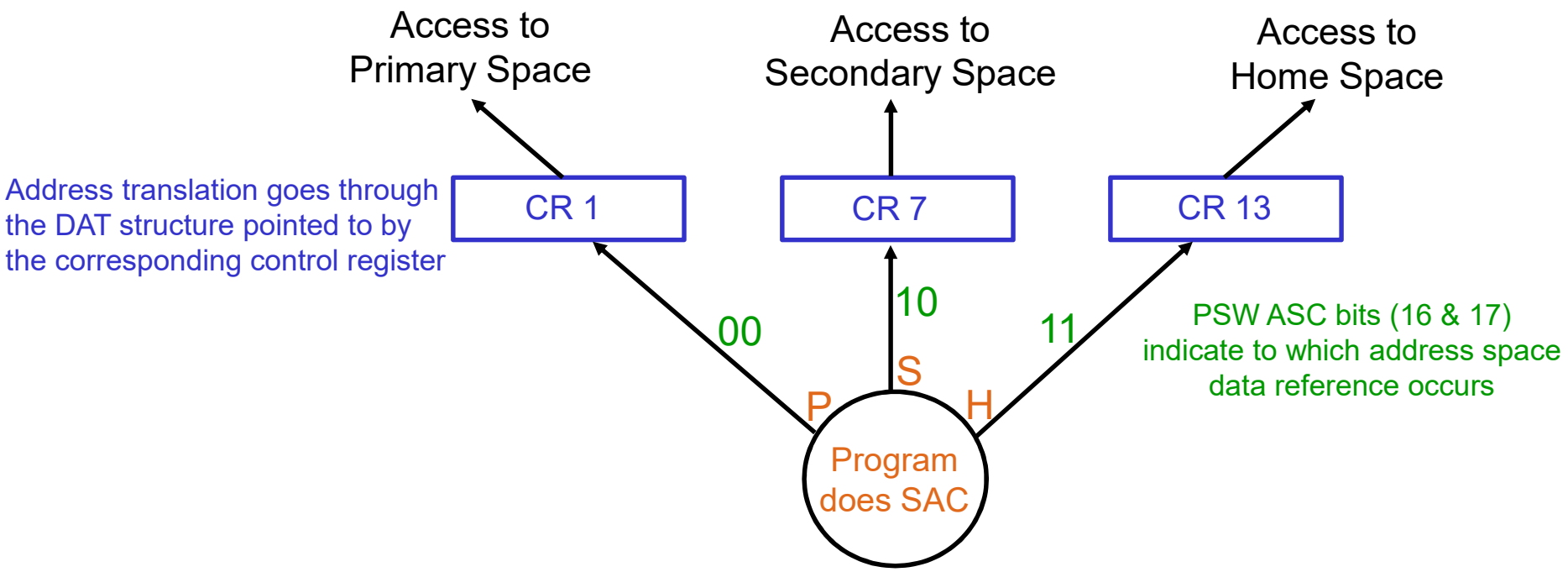
Application: an AR mode program check

```
IEA11054I Access Registers from STCB

  ACCESS REGISTER VALUES
      0-3  00000000  00000000  00000000  00000000
      4-7  00000002  00000000  00000000  00000000
      8-11 00000000  0101006D  00000000  00000000
     12-15 00000000  00000000  0101006D  00000002

ALET TRANSLATION
AR 04 addresses ASID(X'00B3')
AR 09 addresses ASID(X'0011') DSPNAME(SYSZBPX2)
AR 15 addresses ASID(X'00B3')
```

IP SUMM FORMAT report translates some ALETs

APPENDIX

Putting it together (for inquiring minds)



Instruction Fetch is always from the Primary address space! **

Including data spaces in a dump

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

- SLIP or DUMP Command
  - **DSPNAME=**

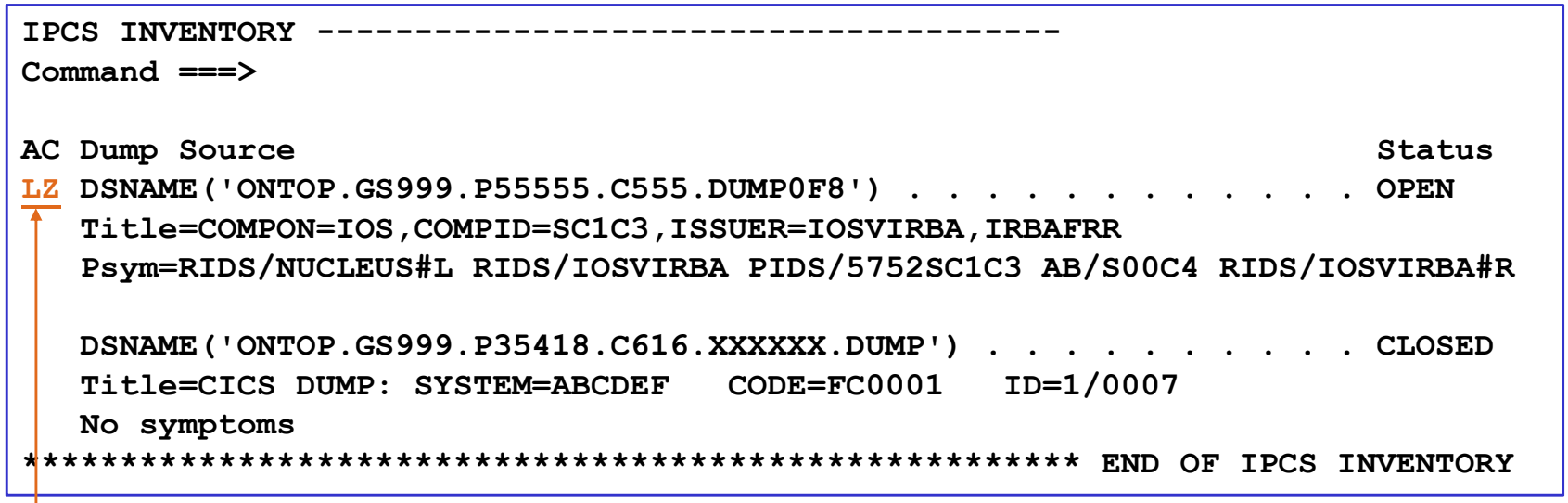
    ex: DUMP COMM=JES2DUMP
    JOBNAME=JES2,DSPNAME=('JES2AUX'.*),etc

    ex: SLIP SET,COMP=0C4,JOBNAME=SAPIAPP,
    JOBLIST=(JES2),DSPNAME=(2B.JES2SAPI),END

- SDUMPX coded macro
  - **DSPLIST**,**LISTD**,**SUMLSTL**

- Standalone dumps
  - AMDSADMP macro to generate SADUMP program
  - Keywords: **dataspaces of asid(x)** or **dataspaces**

Identifying whether data spaces dumped



LZ = list zone

Identifying whether data spaces dumped (continued)

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

LIST ZONE REPORT

```
IPCS OUTPUT STREAM ----------------------------------------------
Command ===>
   ASID(X'0003') DSPNAME(SYSDS000)
     800000.:807FFF. RECORD(89343:89350) POSITIONS(64:4159)
     900000.:906FFF. RECORD(89351:89357) POSITIONS(64:4159)
     A00000.:A02FFF. RECORD(89358:89360) POSITIONS(64:4159)
   X'012000' bytes described in ASID(X'0003') DSPNAME(SYSDS000)

   ASID(X'0003') DSPNAME(SYSDS001)
     800000.:803FFF. RECORD(89361:89364) POSITIONS(64:4159)
     02700000.:02701FFF. RECORD(89365:89366) POSITIONS(64:4159)
     04600000.:04602FFF. RECORD(89367:89369) POSITIONS(64:4159)
   X'9000' bytes described in ASID(X'0003') DSPNAME(SYSDS001)
```

Report shows ranges dumped for each data space.

Identifying data spaces owned by ASID

- **IPCS RSMDATA DSPACE ALL displays all data spaces in the system**
- **IPCS RSMDATA DSPACE ASID(X'yy') displays data spaces owned by ASID yy**

```
JOBNAME  ASID DSP NAME OWNG TCB CUR B MAX B K T S R F TOT R
-------- ---- -------- -------- ----- ----- - - - - - - -----
CONSOLE  000B IEAM02F2 007FD230 00100 00100 0 B A E N 00008
CONSOLE  000B IEEMCS01 007FD230 80000 80000 0 B S E Y 00024
```

See the RSMDATA chapter of *MVS Diagnosis: Reference* for an explanation of this report.

THANK YOU!!!