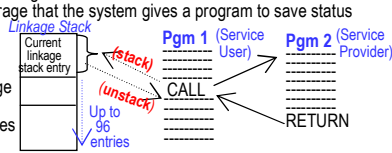


- * **Asynchronous cross memory** communication is a fancy way to describe scheduling an SRB.
 - An SRB is a service request block that a task can schedule to request that some service take place in the same address space or another address space.
 - Any data that the requesting task and the service share must be placed in *common storage* (see #52 zTidbits Storage).
 - SRBs are one way to overlap processing.
 - A task schedules an SRB to perform a service, then continues with its work.
 - When the service completes, it informs the task. The timing, however, is asynchronous; the point when the SRB completes cannot be predicted.

- * **Synchronous cross memory** communication, called cross memory, is both more complex and more flexible than scheduling an SRB.
 - Cross memory requires the programmer to use MVS macros to establish a cross memory environment.
 - This environment defines the authorization requirements that protect the integrity of the address spaces involved.
 - Once this environment is established, the application can use assembler instructions to transfer control from one address space to another.

- Cross memory applications (as well as applications running in a single address space) can use the processor-managed *linkage stack* to simplify program linkages.
 - NOTE** - The *linkage stack* is an area of protected storage that the system gives a program to save status information at a branch (BR) or a program call (PC).
 - The illustration shows how a program uses the linkage stack. The call from Program 1 to Program 2 automatically places all the caller's status on the linkage stack. The return from Program 2 to Program 1 automatically restores all the caller's status and removes the entry from the linkage stack. The linkage stack consist of 2 stacks: the *normal stack* consisting of at least 96 entries for use by pgrams that run under the UOW# and the *recovery linkage stack* which is available to a program's recovery routines after the "stack full" interrupt occurs. More than 96 entries on either stack you can use the **LSEXPAND** macro.
 - In a cross memory environment, the program call (PC) instruction that transfers control to another routine can be either a *basic PC* or a *stacking PC*. (See z/OS MVS Diagnosis Reference)
 - If it is a stacking PC, the system saves status on the linkage stack before it passes control to the PC routine.
 - When the PC routine returns control, the system automatically restores status from the linkage stack.
 - The key fact to remember, however, is that cross memory provides synchronous communication or processing across address spaces.
 - > When a task issues a PC instruction, control passes to the specific PC routine using a hex number.
 - When the PC routine completes, it returns control to the calling routine.



PC number (hexadecimal)	Service description	Component or module
00000000	Linkage index reserve	IEAVLFR
00000001	Linkage index free	IEAVLFR
00000002	Entry table create	IEAVXECR
00000003	Entry table destroy	IEAVXEDE
00000004	Entry table connect	IEAVXECO
00000005	Entry table disconnect	IEAVXEDI
00000006	Authorization index reserve	IEAVXRFE
00000007	Authorization index free	IEAVXRFE
00000008	Authorization index extract	IEAVXRFE
00000009	Authorization index set	IEAVXSET
0000000A	Authorization table set	IEAVXSET
0000000B	PC/AUTH resource manager	IEAVXPAM
0000000C	For use by IBM code only	IEAVXPAX
0000000D	ALESERV ADD/ADDPASN services	IEAVXALA
0000000E	ALESERV DELETE service	IEAVXALD
0000000F	ALESERV EXTRACT/EXTRACTH services	IEAVXALE
00000010	ALESERV SEARCH service	IEAVXALS
00000011	DualPool Router	
00000102	ENQ/DEQ/RESERVE resource termination manager	ISGGTRM1
00000103	Global resource serialization dump services	ISGGC9B0
00000104	Global resource serialization queue scan services (SCOPE is STEP, SYSTEM, or SYSTEMS)	ISGGSC
00000105	Global resource serialization storage management interface	ISGSMI
00000106	Global resource serialization QScan services (SCOPE is LOCAL or GLOBAL)	ISGGSC
00000107	Cross Memory DEQ Service, LINKAGE=SYSTEM	ISGGRT
00000108	Cross Memory ENQ Service, LINKAGE=SYSTEM	ISGGRT

- This mechanism makes use also of the previously existing **PROGRAM CALL** instruction, an extended entry-table entry, and a **PROGRAM RETURN** instruction. The mechanism saves various elements of status, including access-register and general-register contents, during a calling linkage, provides for changing the current status during the calling linkage, and restores the original status during the returning linkage. The linkage stack can also be used to save and restore access-register and general-register contents during a branch-type linkage performed by the instruction BRANCH AND STACK (BAKR).
- A translation mode named *home-space mode* provides an efficient means for the control program to obtain control in the address space, called the *home address space*, in which the principal control blocks for a dispatchable unit (a task or process) are kept.
- PC routine's environment refers to whether the routine runs in supervisor state or problem state.**

- NOTE** - Saving status is a required part of program linkage such as general purpose registers (GPRs), access registers (ARs), the PSW, and other important information. The first thing a program does when it receives control is save the status of its caller. The last thing the program does before it returns control is restore the caller's status. The calling program can then resume processing with its status (including registers and cross memory environment) intact. For example, your stack PC routines might have used the PCLINK **STACK** macro to save a caller's status and then the PCLINK **UNSTACK** macro to restore the status as previously illustrated.
- * In **access register address space control (ASC) mode**, a program can use the full set of assembler instructions (except MVCP and MVCS*) to manipulate data in another address space or in a data space (see #54 zTidbits EA).
- * Unlike basic cross memory (PC), access registers allow *full access to data* in many address spaces or data spaces.
- **ASC mode** determines how the processor resolves address references for the executing program.
- In **primary ASC mode**, the processor uses the contents of general purpose registers to resolve an address to a specific location.

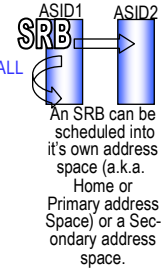
CheatSheet

#58 zTidbits

z/OS Cross Memory

An SRB is a **control block** that represents a routine that performs a particular function or service in a specified address space. An SRB is similar to a TCB in that it identifies a unit of work to the system. Some characteristics of an SRB are:

1. An SRB cannot "own" storage areas. SRB routines can obtain, reference, use, and free storage areas, but the areas must be owned by a TCB.
2. An SRB has associated with it such resources as a dispatchable unit access list (DU-AL), a functional recovery routine (FRR) stack, a linkage stack, and a CPU timer value. The routine that performs the function or service is called the **SRB routine**; initiating the process is called **scheduling an SRB**.



Cross memory allows an application running in one address space to provide services for many users in other address spaces. **SHARE EVERYTHING!**

An SRB represents a **demand** to execute a z/OS service routine on behalf of an end user's request. It is usually initiated by the system code executing in one address space to perform an action affecting the same or another address space. Thus a function in one address space can schedule an SRB to prompt a function in another address space to process some data place in common storage by the first function. As an example, when VTAM has received a message from a terminal via the I/O subsystem, it places the message in common storage, determines which address space it is intended for, then schedules an SRB into that address space to inform it that a message has been received and where it is located in storage.

- UOW - Unit of work
- MVCP (MOVE TO PRIMARY) instruction
- MVCS (MOVE TO SECONDARY) instruction
- PSAAOLD : A pointer to the Address Space Control Block (ASCB) of the address space currently scheduled on this CP. The ASCB holds basic information about an address space, including Jobname and Address Space ID.

- * In access register ASC mode, an access register (AR) identifies the space the processor is to use to resolve an address.
 - The processor uses the contents of an AR as well as the contents of general purpose registers to resolve an address to a specific location.
- * In AR ASC mode, a program can move, compare, or perform operations on data in other address spaces or in data spaces.
 - NOTE:** ARs do not enable a program to transfer control from one address space to another. That is, you cannot use ARs to transfer control from a program in one address space to a program in another address space. For that, you need cross memory. You *can*, however, use ARs without using cross memory.
 - If your application needs to manipulate data in other address/data spaces but does not need to transfer control to other address spaces, use ARs.
 - If your application needs to transfer control to routines in other address spaces but does not need to manipulate data, use cross memory.
 - If your application needs both the transfer of control and the manipulation of data, use *both* cross memory and ARs.

When Should You Use Synchronous Cross Memory Communication?

- The use of synchronous cross memory communication to provide services to users 'can' provide virtual storage constraint relief as well as improve the integrity of the service and its data. [no duplication of services]
- Consider using synchronous cross memory communication if you wish to:
 1. Isolate the service and its data from the user of the service
 2. Make the service available to multiple users without the need to store it in commonly addressable storage
 3. Replace an existing service request block (SRB) routine to gain improved performance or simplify communication
 4. Provide an authorized service to problem state programs
- * Synchronous cross memory communication enables you to provide services to many users without making the service available in commonly addressable storage.
 - At the same time, you can isolate the service from the user, thus protecting it by having the service in its own address space.
 - To access data that is in another address space or in a data space, or to store data into another address space or data space, **IBM recommends** that the PC routine use ARs.
 - To use ARs, the PC routine must be in AR ASC mode.
 - A PC routine can use ARs in the same way any other program uses them.
 - The PC routine, if necessary, can access data in the user's address space without using ARs by using MVCP or MVCS instructions. *Note: the service provider must have obtained SSAR authority (set secondary ASN authorization).*

Terms and Definitions

- Address space control (ASC) mode:** The mode (determined by the PSW) that tells the system where to find the data it is to reference. Two ASC modes are AR and primary. For each ASC mode, the following table defines:
 - The address space from which the system fetches instructions
 - The address space or data space that the system accesses when an instruction, other than an MVCP or MVCS instruction, references data
 - The address space the system accesses when an MVCP or MVCS instruction references data
- AR ASC mode:** The ASC mode in which the system uses both the GPR (used as the base register) and the corresponding AR to resolve an address in an address/data space.
- Basic PC:** Transfers control to another program, the PC routine. The basic PC requires the service provider to save and restore the user's environment. The PC routine can be in the same address space as the program that issues the PC instruction, or in a different address space.
- Cross memory local (CML) lock:** The LOCAL lock of an address space other than the home address space.
- Cross memory mode:** Cross memory mode exists when at least one of the following conditions are true:
 - The primary address space (PASN) and the home address space (HASN) are different address spaces.
 - The secondary address space (SASN) and the home address space (HASN) are different address spaces.
 - The ASC mode is secondary.
- Home address space:** The address space in which MVS initially dispatches a TCB^s or SRB (work unit). In the case of a TCB, the home address space contains the TCB. PSAAOLD# points to the home address space. When z/OS initially dispatches a work unit, the home address space, the primary address space, and the secondary address space are all the same. During execution of the work unit, the home address space remains the same. The primary and secondary address spaces may be changed, however, through the PC, PR, PT, or SSAR instructions.
- Primary address space:** The address space whose segment table is used to fetch instructions in primary, secondary, and AR ASC modes. A program in primary mode fetches data from the primary address space.
- Primary ASC mode:** The ASC mode in which the system uses the GPRs, but not the ARs, to resolve an address in an address space. In primary ASC mode, the system fetches instructions and data from the primary address space.
- PC number:** A number that identifies a PC routine. The service provider creates the number, by using MVS services and supplies it to the user. The user specifies the number in a PC instruction to identify the PC routine that is to be invoked.
- PC routine:** A program that receives control as the result of a PC instruction's executing and performs a service for the caller.
- Secondary address space:** The address space whose segment table the system uses to access data in secondary ASC mode.
- Secondary ASC mode:** The ASC mode in which the system fetches instructions from the primary address space and data from the secondary address space.
- Space switch routine:** A program that issues a PC instruction that causes the primary address space to change.
- Stacking PC:** Transfers control to another program, the PC routine. The stacking PC uses the linkage stack for storing the caller's status. It provides more options and more automatic function than the basic PC instruction. The PC routine can be in the same address space as the program that issues the PC instruction, or a different address space. **IBM recommends** using the stacking PC instead of the basic PC.
- About SRBs:** They're created using a different routine than a TCB ATTACH. They are initiated using a SCHEDULE routine (macro) and are run in a privileged state known as Supervisor State (executing in key 0). There are two types of SRB: SRBs with global priority, which have a very high dispatching priority for inter-address space services and are given priorities above that of any address space, regardless of the actual address space in which they are dispatched. SRBs with local priority takes on the dispatch priority of the address space in which they are scheduled to run (**intra-address space**).

TCB - see #10 zTidbits (Dispatchable UOW)