Stack Organisation

A useful feature that is included in the CPU of most computers is a stack or last-in, first-out (LIFO) list. A stack is a storage device that stores information in such a manner that the item stored last is the first item retrieved. The stack in digital computers is essentially a memory unit with

1. **Address register** that can count only (after an initial value is loaded into it).
2. **Stack Pointer register**: It holds the address for the stack because its value always points at the top item in the stack.

The two operations of a stack are the insertion and deletion of items. The operation of insertion is called push (or push-down) because it can be through of as the result of pushing a new item on top. The operation of deletion is called pop (or pop-up) because it can be thought of as the result of removing one item so that the stack pops up. However, nothing is pushed or popped in a computer stack these operations are simulated by incrementing or decrementing the stack pointer register.

Various Stack organisation that can be employed in Control Unit are

1. **Register Stack:** A stack can be placed in portion of a large memory or it can be organized as a collection of a finite number of memory words or registers. Following figure shows the organization of a 64-word register stack. The stack pointer register SP contains a binary number whose value is equal to the address of the word that is currently on top of the stack. Three items are placed in the stack: A, B, and C, in that order. Item C is on top of the stack so that the content of SP is now 3. To remove the top item, the stack is popped by reading the memory word at address 3 and decrementing the content of SP. Item B is now on top the stack since SP holds address 2. To insert a new item, the stack is pushed by incrementing SP and writing a word in the next-higher location in the stack. Note that item C has been read out but not physically removed. This does not matter because when the stack is pushed, a new item is written in its place.
2. **Memory Stack:** A stack can exist as a stand-alone unit or can be implemented in a random-access memory attached to a CPU. The implementation of a stack in the CPU is done by assigning a portion of memory to a stack operation and using a processor register as a stack pointer. Following figure (after Register Stack figure) shows a portion of computer memory partitioned into three segments: program, data, and stack. The program counter PC points at the address of the next instruction in the program. The address register AR points at an array of data. The stack pointer SP points at the top of the stack. The three registers are connected to a common address bus, andeither one can provide an address for memory. PC is used during the fetch phase to read aninstruction. AR is used during the execute phase to read an operand. SP is used to push or popitems into or from the stack

Compiled by: Dr. Sandeep Maan, Assoc Prof in Computer Sc, GCG Sec-14, Gurugram.
Reference: M Morris Mano, Computer System Architecture, Third Edition, Pearson Education

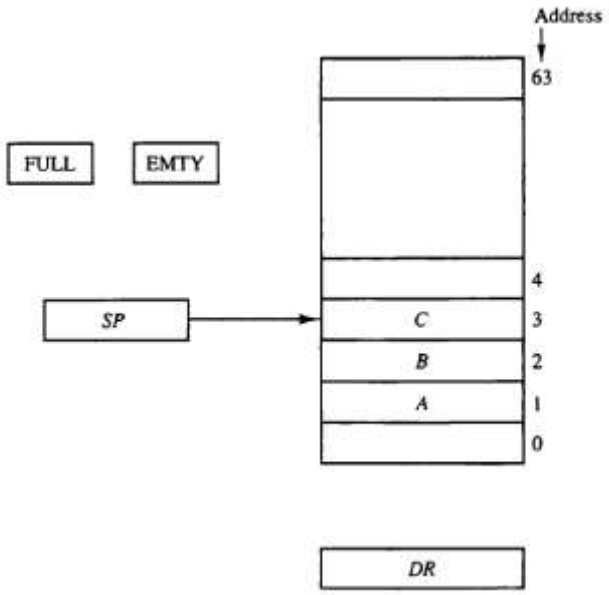

Figure 1: Implementing 64-word Register Stack

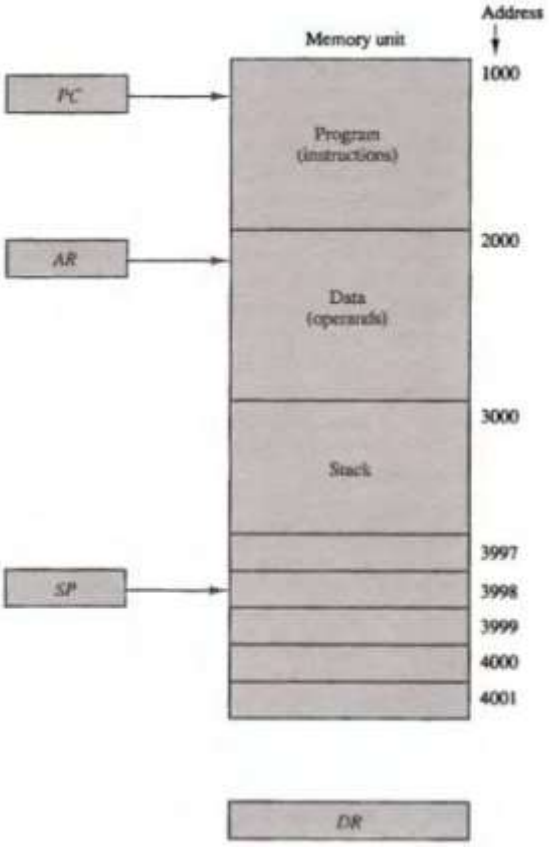

Figure 2: Implement Memory Stack

Most computers do not provide hardware to check for stack overflow (full stack) or underflow (empty stack). The stack limits can be checked by using two processor registers: one to hold the

upper limit and the other to hold the lower limit. After a push operation, SP is compared with the upper-limit register and after a pop operation, SP is compared with the lower-limit register.
The two microoperations needed for either the push or pop are (1) an access to memory through SP, and (2) updating SP. Which of the two microoperations is done first and whether SPis updated by incrementing or decrementing depends on the organization of the stack.

NOTE: Students are suggested to study Reverse Polish Notation & Equation Solutions topics also