

Computer System Architecture
COMP201Th
Unit: 2
Basic Computer Organization and Design

Lecture: 7

Stack Organization

A stack is an ordered linear list in which **all insertions and deletions are made at one end, called top**. It uses **Last In First Out (LIFO)** access method which is the most popular access method in most of the CPUs.

A register is used to store the address of the topmost element of the stack which is known as Stack Pointer(SP) because its value always points at the top item in the stack.

The main two operations that are performed on the operands of the stack are Push and Pop. These two operations are performed from one end only.

- **Push Operation:** The operation of **inserting an item** onto a stack is called push operation.
- **Pop Operation:** The operation of **deleting an item** onto a stack is called pop operation.

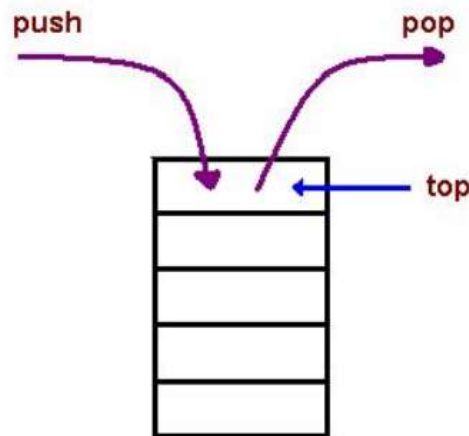


Figure 1 Stack

Applications:

- **Evaluation of mathematical expressions** using Reverse Polish Notation.
- To **reverse a word**. A given word is pushed to stack-letter by letter-and then popped out letters from the stack.
- An **undo mechanism in text editors**; this operation is accomplished by keeping all text changes in a stack.
- Backtracking: this is a process when it is required to access the most recent data element in a series of elements.
- Language processing

- space for parameters and local variables is created internally using a stack.
- The compiler's syntax check for matching braces is implemented by using stack.

There are two types of stack organization which are used in the computer hardware:

- **Register stack:** It is built using register
- **Memory stack:** It is logical part of memory allocated as stack. The logically partitioned part of RAM is used to implement stack.

Register stack:

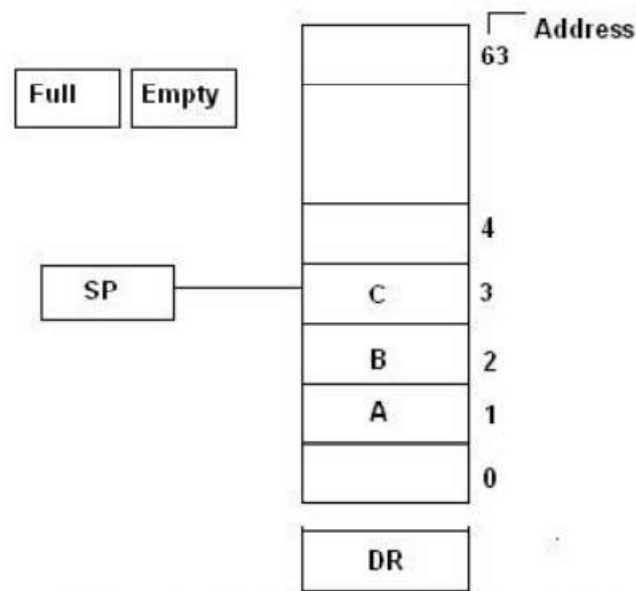


Figure 2: a. 64 word Register stack

As shown in fig 2, there are 64 registers used to make a register stack. The numbers 0,1,2,3,.....upto 63 denote the address of different registers.

- SP is a pointer which points to the top of the stack i.e. it currently points to the item at the top.
- The two more registers called FLAG and EMPTY are used. These are made up of flip-flops. It indicates whether the stack is full or not.
 - If FULL = 1, then EMPTY = 0 → stack is full.
 - If FULL= 0, then EMPTY =1 → stack is empty.
- DR is the data register through which data is transferred to and from the stack.

Zero address instructions are used in registers stack organization i.e. the address that does not contain the address of the operands.

In 64-word stack, the stack pointer contains 6 bits because $2^6=64$.

Since, SP has only 6 bits, it cannot exceed a number greater than 63(111111 in binary). When 63 is incremented by 1 the result is 0 since $111111+1=1000000$, but SP can accommodate only the six least significant bits \rightarrow SP points to 000000 address register which implies stack is full.

The PUSH operation is implemented with the following sequence of microoperations:

$SP \leftarrow SP+1$	Increment stack pointer
$M[SP] \leftarrow DR$	Write item on top of the stack
If (SP=0) then (FULL \leftarrow 1)	Check if stack is full
EMPTY \leftarrow 0	Mark the stack not empty

- The stack pointer is incremented so that it points to the address of the next higher word.
- A memory write operation inserts the word from DR into the top of the stack.
 - SP holds the address of the top of the stack and that $M[SP]$ denotes the memory word specified by the address presently available in SP.
- The first item stored in the stack is at address 1. The last item is stored at address 0.
- If SP reaches 0 \rightarrow stack is full of items, so full is set to 1.
- If an item is written in the stack, obviously the stack cannot be empty, so EMPTY is cleared to 0.

A new item is deleted from the stack if the stack is not empty (if EMPTY=0). The POP operation consists of following sequence of microoperations:

$DR \leftarrow M[SP]$	Read item from the top of the stack
$SP \leftarrow SP-1$	Decrement stack pointer
If (SP=0) then (EMPTY \leftarrow 1)	Check if stack is empty
FULL \leftarrow 0	Mark the stack not full

- The top item is read from the stack into DR.
- The stack pointer is then decremented.
- If its value reaches zero, the stack is empty, so EMPTY is set to 1. This condition is reached if the item read was in location 1.

Memory Stack: The RAM is divided into three logical parts:

- **Program:** The logical part of RAM where programs are stored.
- **Data:** It is the logical part of the RAM where data(operands) are stored.
- **Stack:** It is the part of RAM used to implement stack.

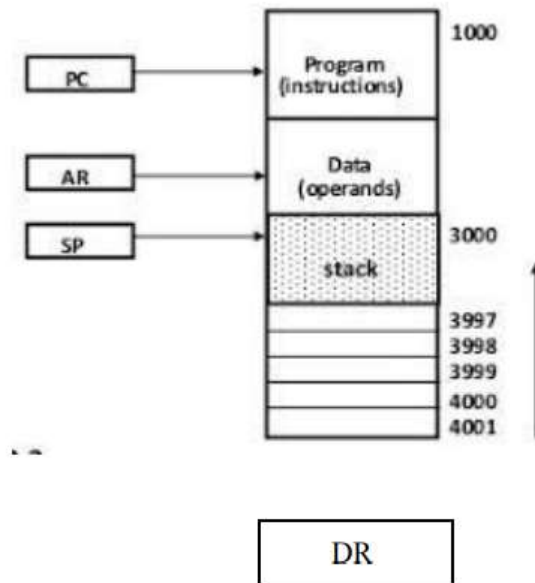


Figure 3: Memory Stack

Data register is used to store data. Data is pointed by the address pointer or register. Program Counter (PC) is used to point the program instruction.

As shown in the diagram, the initial value of SP is 4001 and the stack grows with decreasing addresses.

A new item is inserted with the PUSH operation as follows:

$$SP \leftarrow SP - 1$$

$$M[SP] \leftarrow DR$$

A new item is deleted with a POP operation as follows:

$$DR \leftarrow M[SP]$$

$$SP \leftarrow SP + 1$$

Application- Evaluation of mathematical expressions using Reverse Polish Notation:

Computer evaluates the mathematical expressions by representing it in reverse polish notation and stack is used for evaluation as shown below:

Infix	Prefix or Polish.	Postfix or reverse Polish notation
A + B	+AB.	AB+

Example ∴ (3 * 4) * (5 * 6). 3 4* 5 6* *

						→	6				
→	4			→	5		5	→	30		
3	3	→	12		12		12		12	→	360