Stack, Stack pointer and Subroutines in 8085 – With coding examples

The stack is a reserved area of the memory in RAM where we can store temporary information. Interestingly, the stack is a shared resource as it can be shared by the microprocessor and the programmer. The programmer can use the stack to store data. And the microprocessor uses the stack to execute subroutines. The 8085 has a 16-bit register known as the ‘Stack Pointer.’

This register’s function is to hold the memory address of the stack. This control is given to the programmer. The programmer can decide the starting address of the stack by loading the address into the stack pointer register at the beginning of a program.

The stack works on the principle of First In Last Out. The memory location of the most recent data entry on the stack is known as the Stack Top.

How does a stack work in assembly language?

We use two main instructions to control the movement of data into a stack and from a stack. These two instructions are PUSH and POP.

PUSH – This is the instruction we use to write information on the stack.
POP – This is the instruction we use to read information from the stack.

There are two methods to add data to the stack: Direct method and Indirect method

Direct method

In the direct method, the stack pointers address is loaded into the stack pointer register directly.

```
LXI SP,8000H
```

```
LXI H,1234H
PUSH H
POP D
HLT
```

Explanation of the code

LXI SP, 8000H – The address of the stack pointer is set to 8000H by loading the number into the stack pointer register.

LXI H, 1234H – Next, we add a number to the HL pair. The most significant two bits will enter the H register. The least significant two bits will enter the L register.

PUSH H – The PUSH command will push the contents of the H register first to the stack. Then the contents of the L register will be sent to the stack. So the new stack top will hold 34H.

POP D – The POP command will remove the contents of the stack and store them to the DE register pair. The top of the stack clears first and enters the E register. The new top of the stack is 12H now. This one clears last and enters the D register. The contents of the DE register pair is now 1234H.

HLT – HLT indicates that the program execution needs to stop.

Indirect method

In the indirect method, the stack pointers address is loaded into the stack pointer register via another register pair.

```
LXI H,8000H
SPHL
LXI H,1234H
PUSH H
POP D
HLT
```

Explanation of the code

LXI H, 8000H – The number that we wish to enter into the stack pointer, 8000H, is loaded into the HL pair register.

SPHL – This is a special command that we can use to transfer data from HL pair to Stack pointer (SP). Now, the contents of the HL pair are in the SP.

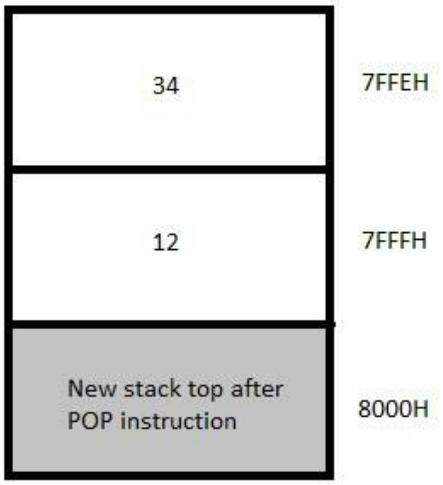
LXI H, 1234H – Next, we add a number to the HL pair. The most significant two bits will enter the H register. The least significant two bits will enter the L register.

PUSH H – The PUSH command will push the contents of the H register first to the stack. Then the contents of the L register will be sent to the stack. So the new stack top will hold 34H.

POP D – The POP command will remove the contents of the stack and store them to the DE register pair. The top of the stack clears first and enters the E register. The new top of the stack is 12H now. This one clears last and enters the D register. The contents of the DE register pair is now 1234H.

HLT – HLT indicates that the program execution needs to stop.

Both the methods can be shown diagrammatically with the following diagram.



What is a Subroutine is assembly language?

A subroutine is a small program written separately from the main program to perform a particular task that you may repeatedly require in the main program. Essentially, the concept of a subroutine is that it is used to avoid the repetition of smaller programs.

Subroutines are written separately and are stored in a memory location that is different from the main program. You can call a subroutine multiple times from the main program using a simple CALL instruction.

What are the conditional CALL statements in assembly language?

You can use conditional CALL statements, too, according to your needs. These statements enter a subroutine only when a certain condition is met.

CC	Call at address if cy (carry flag) = 1
CNC	Call at address if cy (carry flag) = 0
CZ	Call at address if ZF (zero flag) = 1
CNZ	Call at address if ZF (zero flag) = 0
CPE	Call at address if PF (parity flag) = 1
CPO	Call at address if PF (parity flag) = 0
CN	Call at address if SF (signed flag) = 1
CP	Call at address if SF (signed flag) = 0

The subroutine can be exited from using a return (RET) instruction.

What are the conditional return (RET) statements in assembly language?

Akin to the CALL instruction, we have conditional RET statements too. These statements ensure that a subroutine is exited only when a certain condition is met.

RC	Return from subroutine if cy (carry flag) = 1
RNC	Return from subroutine if cy (carry flag) = 0
RZ	Return from subroutine if ZF (zero flag) = 1
RNZ	Return from subroutine if ZF (zero flag) = 0
RPE	Return from subroutine if PF (parity flag) = 1
RPO	Return from subroutine if PF (parity flag) = 0

RN Return from subroutine if SF (signed flag) = 1
RP Return from subroutine if SF (signed flag) = 0

After the completion of the subroutine, the main program begins from the instruction immediately following the CALL instruction.

What are the advantages of using subroutines?

- Subroutines avoid the repetition of instructions.
- They give an aspect of modular programming to the entire program.
- Improves efficiency by reducing the size of the memory needed to store the program.

The PUSH Instruction:

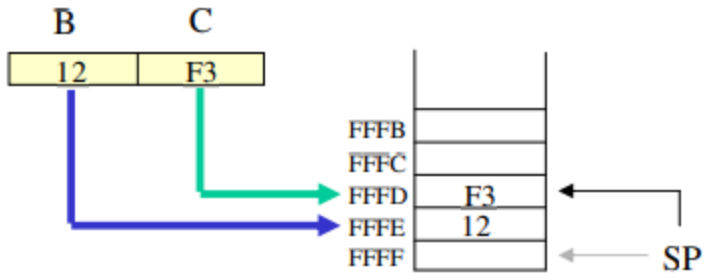
PUSH B (1 Byte Instruction)

– Decrement SP

– Copy the contents of register B to the memory location pointed to by SP

– Decrement SP

– Copy the contents of register C to the memory location pointed to by SP



The POP Instruction:

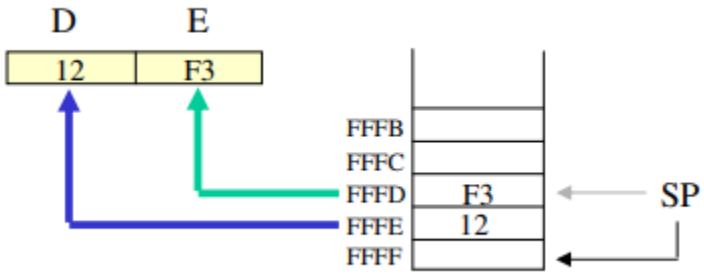
POP D (1 Byte Instruction)

– Copy the contents of the memory location pointed to by the SP to register E

– Increment SP

– Copy the contents of the memory location pointed to by the SP to register D

– Increment SP



The PSW Register Pair

The 8085 recognizes one additional register pair called the PSW (Program Status Word).

– This register pair is made up of the Accumulator and the Flags registers.

It is possible to push the PSW onto the stack, do whatever operations are needed, then POP it off of the stack.

– The result is that the contents of the Accumulator and the status of the Flags are returned to what they were before the operations were executed.