VIVEKANANDA COLLEGE
THAKURPUKUR
KOLKATA-700063

NAAC ACCREDITED 'A' GRADE

Topic: stack and sub- routine

Course Title: microprocessor

Paper: ELTG-CC-4-4-TH

Unit: microprocessor and microcontroller

Semester: fourth

Name of the Teacher: Anjan Paul

Name of the Department: Electronic science

Sub – routine in 8085:

Sub – routine is the special part of the programs which is written outside the main programs. Subroutine is written because while preparing a program , a condition may arise where a group of instruction repeatedly appears ,so instead of writing this group repeatedly , it can be written separately from the main program. So any part of main program which is used repeatedly can be written in the subroutine program. Any subroutine call by **CALL** instruction and by using return (**RET**) instruction we return from subroutine to main program. Subroutine with in a subroutine is known as the Nested subroutine.

As after the completion of the subroutine we have to come back to the main program, for coming back to the main program we require the address of the memory location from where the mpu has to resume its work. While going to the subroutine in response to a CALL instruction , the mpu stores the returning address on the top of the stack. When the mpu executes a RET instruction , the returning address is taken from the top of the stack.

CALL instruction:

It is 3 byte instruction. This instruction is used to jump from the main program to the subroutine program. The starting address of the subroutine is specified in instruction that is CALL 16 bit address. For ex if we written CALL 9000H then the subroutine will start at 9000H memory location. There are 4 step in the implement of CALL instruction

1> When a CALL instruction is executed the first step is to save the returning address on the top of the stack.

2> Then mpu jumps to the specified CALL location where the subroutine is present.

3> Execute the subroutine and subroutine is terminated by RET

instruction. When the mpu executes the RET instruction the mpu will fetch the returning address from the top of the stack.

4> After getting the address form top of the stack, the mpu resumes its execution in the main program.

RET instruction:

it is 1 byte instruction. It is used to return from subroutine to main program.

Q> why stack pointer locate in last memory location?

Ans. To avoid data collision.

Stack and subroutine in 8085.

Stack is a group of memory location. It is used for storage of information temporary during execution of programs. Stack follows LIFO(last input fast output). The name " stack" is given to this memory because the data to be stored at the top of the memory location just like a stacking. As we cannot pull out an item from a stack, similarly we cannot pop up a data from the middle of the stack memory. The data can be popped out only from the top of the stack memory.

Stack pointer initialize:

The beginning of stack pointer is defined by using the instruction,

LXI SP, 16 bit address which load the 16 bit memory address in the stack pointer of the mpu. [SP denote stack pointer].

Data entry process in stack pointer:

Once the stack location is defined then the storing of data bytes begins at the memory address that is one less than the address of the stack pointer. For example if the stack pointer is loaded with the memory address 9000H (LXI SP, 9000H), then storing of data bytes begins at 8999H and continues in reversed numerical order (decreasing memory address such as 8998H, 8997H etc).

Instruction used in stack pointer:

The programmer can store and retrieved the content of a register pair by using **PUSH** and **POP** instruction. PUSH instruction copies the content of register pair into the stack. For every PUSH instruction = decrement stack pointer + PUSH register pair. POP

instruction used for retrieved the data stored in the stack. For every POP instruction = increment stack pointer + retrieved 8 bit data.

Opcode operand

LXI SP, 16 bit address [load stack pointer with 16 bit address]

PUSH R_P [store register pair on stack]

PUSH B [this is 1 byte instruction . it copies the content of BC register pair on the stack . first the stack pointer is decremented and the content of the higher order register (reg B) are copied into that location. Now stack pointer register again decremented and content of the lower order register(reg C) are copied into that location.]

PUSH D

PUSH H

PUSH PSW (accumulator + flag). [this last 3 instruction is similar to PUSH B]

POP R_P [retrieve register pair from stack]

POP B [this is 1 byte instruction . it copies the content of top two memory location of the stack into the specified register pair. First the content of the memory location indicated by the stack pointer are copied into lower order register (reg C) and stack pointer register is incremented by 1. Then content of the next memory location are copied into the higher order register

(reg B) and the stack pointer is again incremented by 1.]

POP H

POP D

POP PSW (A+ F) [this last 3 instruction is similar to P
B]