Title of the course is “Microcontroller Architecture and Programming”.

The module name is “Push and Pop *opcodes*”.

I am Prashant Chodankar, working as Associate Professor of Physics at Government, College of Arts, Science & Commerce, Khandola.

These are outline of this module:

1. Introduction
2. Push and Pop *opcodes*

The Learning Outcomes of the module are:

At the end of the module, learner will be able to:

- Describe how data may be pushed and popped using a stack.
- Write simple programs to temporarily store and retrieve data.

In last module we have seen how the data can be moved from source to destination, by using *opcode,* MOV

The PUSH and POP *opcodes* specify the direct address of the data to be copied. The data moves between an area of internal RAM, known as the stack, and the specified direct address and the vice a versa. The stack pointer (a special function register) represented by (SP), contains the address of the RAM where data *from* the source address to be {moved} PUSHed, or from where the data to be { re

shifted} POPed *to* the destination address {when it is found} . The SP register actually is used in the indirect addressing mode but is not named in the mnemonic. It is implied that the SP holds the indirect address, whenever **PUSHing** or **POPing** is taking place.

A PUSH *opcode* copies data from the source address to the stack area. SP is incremented by one before the data is copied to the internal RAM location contained in SP so that the data is stored from low addresses to high addresses in the internal RAM.

Example on the display has the stack address, SP = 20h. Data from address location 01h is copied at stack area location 21h. Likewise when second PUSH instruction is executed, Data from address location 02h is copied at stack area location 22h and so on. Therefor the stack grows up in memory as it is PUSHed. Excessive PUSHing can make the stack exceed 7Fh (the top of internal RAM), after which PUSHed data is lost. {In the given example we can push the data 4 times as only 4 stack locations are present on the display.}

A POP *opcode* copies data from the stack to the destination address {in the reversed manner as the PUSH instruction}. SP is decremented by one after data is copied from the stack RAM address to the direct destination to ensure that data placed on the stack is retrieved in the same order as it was stored.

The PUSH and POP opcodes behave as explained in the table displayed on the screen.

	Mnemonic	Operation
1	PUSH add	Increment SP; copy the data in “add” location to the internal RAM address contained in SP
2	POP add	Copy the data from the Internal RAM address contained in SP to “add” location; And then decrement the SP

The SP register is set to 07h when the 8051 is reset, which is the same direct address in internal RAM as register R7 in bank 0. The first PUSH *opcode* would write data to R0 of bank 1. {since PUSH instruction will increment SP}

The SP should be initialized by the programmer to point to an internal RAM address above the highest address likely to be used by the program as temporary storage.

The following table shows examples of PUSH and POP o*pcodes*:

	Mnemonic	Operation
1	MOV 81h, #30h	Copy the immediate data 30h to the SP. Note here that, 81h is the Address of SFR, SP. And now stack pointer is pointing at 30h
2	MOV R0, #0ACh	Copy the immediate data ACh to R0 Note: address of R0 = 00h(Bank 0)
3	PUSH 00h	Increments the stack pointer, i.e. SP = 31h; Address 31h contains the data ACh
4	PUSH 00h	SP = 32h; address 32h contains the data ACh.
5	POP 01h	register R1 now contains the number ACh SP = 32-1 = 31h; Since address of R1(Bank 0) = 01h
6	POP 80h	port 0 latch now contains the data = Ach, of location 31; SP decrements to 30h:

Caution here that, when the SP reaches FFh it rolls over to 00h (R0).

RAM ends at address 7Fh; PUSHes above 7Fh result in errors.

Note that:

1) The direct addresses, *not* register names, must be used for R0-R7. The stack mnemonics have no way of knowing which bank is in use.

2) The SP is usually set at addresses above the register banks.

These are the books which I have referred.

Thank you very much.