Title of the course is “Microcontroller Architecture and Programming”

The module name is “Data movement, external data move, code memory read only data moves”.

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. External data Moves
2. Code Memory Read Only Data Moves

The Learning Outcomes of the module are:

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

- Use commands that place data in external memory.
- Use commands that get data from ROM addresses.
- Write simple data movement programs.

1. External Data Moves

We know that one can access only 256 bytes of RAM & 4K of ROM with 8051chip. It is possible to expand RAM and ROM memory space by adding external memory chips to the 8051 microcontrollers. The external memory can be as large as 64K for each of the RAM and ROM memory areas. *Opcodes* that access this external memory always use indirect addressing to specify the external memory.

Table on the screen shows that registers R0, R1 and the suitably named DPTR_can be used to hold the address of the data byte in external RAM. R0 and R1 are limited to external RAM address ranges from 00h to 0FFh, (256 Bytes data) while the DPTR register, can address the maximum RAM space of 0000h to 0FFFFh. (65536 bytes= 64K)

An X is added to the MOV mnemonics to serve as a reminder that the data move is external to the 8051.

	Mnemonic	Operation
1	MOVX A, @Rp	Copy the contents of the external address in register, Rp to Accumulator
2	MOVX A, @DPTR	Copy the contents of the external address in DPTR to A
3	MOVX @Rp, A	Copy data from Accumulator to the external address in Rp
4	MOVX @DPTR, A	Copy data from A to the external address specified in DPTR

Let us see the examples of external moves using register and indirect addressing modes:

	Mnemonic	Operation
1	MOVX @DPTR, A	Copy data from register A to the 16-bit address in DPTR

2	MOVX @R0, A	Copy data from A to the 8-bit address in reg. R0
3	MOVX A, @R1	Copy data from the 8-bit address In R1 to A
4	MOVX A, @DPTR	Copy data from the 16-bit address in DPTR to A

Note that:

a. All external data moves must involve the A register.
b. R0 or R1 can address 256 bytes; DPTR can address 64K bytes.
c. MOVX is normally used with external RAM or I/0 addresses.
d. Note that there are two sets of RAM addresses between 00h and 0FFh: one internal and one external to the 8051.

---

2. Code Memory Read-Only Data Moves

Data moves between RAM locations and 8051 registers are made by using MOV and MOVX *opcodes*. The data is usually of a temporary or "scratch pad" nature, and disappears when the system is powered down.

While programming, many times, an access to a pre-loaded mass of data is needed, such as when using tables of predefined values etc. (ex. Capital cities of different states in the country). This data must be permanent, to be of repeated use and is stored in the program ROM, externally. Access to this data is made possible by using indirect

addressing and the A register in combination with either the PC or the DPTR. In both cases, the number in register A is added to the pointing register to form the address in ROM where the desired data is to be found. The data is then fetched from the ROM address so formed and placed in the Accumulator. The original data in A is lost, and the addressed data takes its place.

The letter C is added to the MOV mnemonic to highlight the use of the *opcodes* for moving data from the source address in the code ROM to the A register in the 8051. Here are the mnemonics for external code data movement on the screen.

	Mnemonic	Operation
1	MOVC A, @A+DPTR	Copy the code byte/data, found at the ROM address formed by adding A and the DPTR, to A
2	MOVC A, @A+PC	Copy the code byte, found at the ROM address formed by adding A and the PC, to A

Note that the DPTR and the PC are not changed; the A register contains the ROM code byte found at the address, so formed.
Let us see the examples of code ROM moves using register and indirect addressing modes:

	Mnemonic	Operation
1	MOV DPTR, #1234h	Copy the immediate number 1234h to the DPTR
2	MOV A, #05h	Copy the immediate number 05h to A
3	MOVC A, A+DPTR	Copy the contents of address formed by adding DPTR with A. i.e. 1234 + 5 = 1239h, location to A. Note that accumulator will change from 05h to code byte 1239h
4	MOV A, #05h	Copy the immediate number 05h to A
5	MOVC A,@A+PC	Assume that PC= 4000h. Copies the contents of address formed by adding 4000h to 5h = 4006h to A.

Value of DPTR or PC is added to A to form the final address, and the code byte data will then be copied to Accumulator .

Note that:

I. The PC is incremented by 1 (to point to the next instruction) before it is added to A to form the final address of the code byte.
II. All data is moved, from the code memory to the A register.
III. MOVC is normally used with internal or external ROM and can address 4K of internal OR 64K of external code.

The books referred for this module are listed.

Thank you very much.