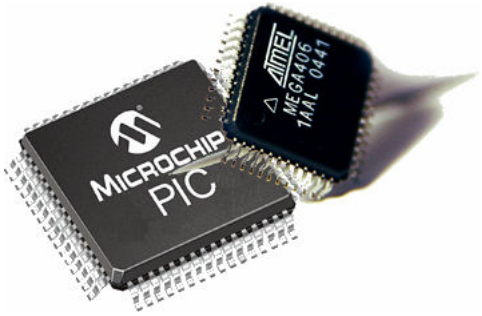
ADDRESSING MODES



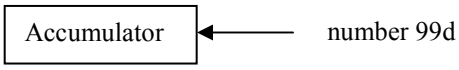
ADDRESSING MODES

There are a number of addressing modes available to the 8051 instruction set, as follows:

Immediate Addressing	Register Addressing	Direct Addressing
Indirect Addressing	Relative Addressing	Absolute addressing
Long Addressing	Indexed Addressing	

Immediate Addressing
Immediate addressing simply means that the operand (which immediately follows the instruction op. code) is the data value to be used. For example the instruction:

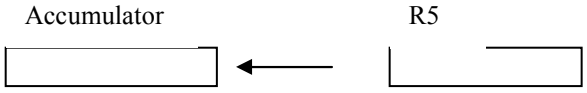
MOV A, #99d



Moves the value 99 into the accumulator (note this is 99 decimal since we used 99d). The # symbol tells the assembler that the immediate addressing mode is to be used.

Register Addressing
One of the eight general-registers, R0 to R7, can be specified as the instruction operand. The assembly language documentation refers to a register generically as *Rn*. An example instruction using register addressing is :

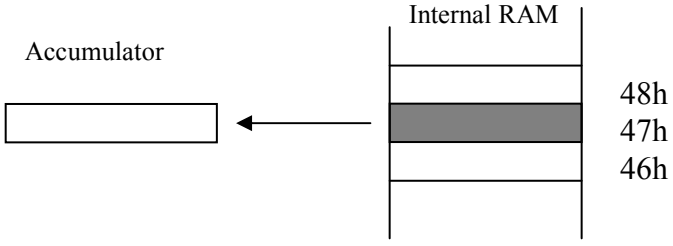
ADD A, R5 ; Adds register R5 to A (accumulator)



Here the contents of R5 is added to the accumulator. One advantage of register addressing is that the instructions tend to be short, single byte instructions.

Direct Addressing
Direct addressing means that the data value is obtained directly from the memory location specified in the operand. For example consider the instruction:

MOV A, 47h

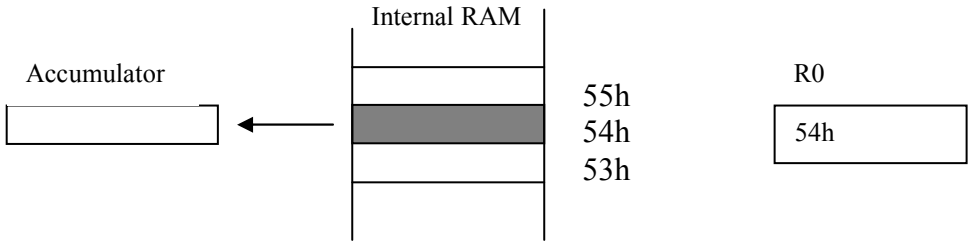


The instruction reads the data from Internal RAM address 47h and stores this in the accumulator. Direct addressing can be used to access Internal RAM , including the SFR registers.

Indirect Addressing

Indirect addressing provides a powerful addressing capability, which needs to be appreciated. An example instruction, which uses indirect addressing, is as follows:

MOV A, @R0



Note the @ symbol indicated that the indirect addressing mode is used. R0 contains a value, for example 54h, which is to be used as the address of the internal RAM location, which contains the operand data. Indirect addressing refers to Internal RAM only and cannot be used to refer to SFR registers.

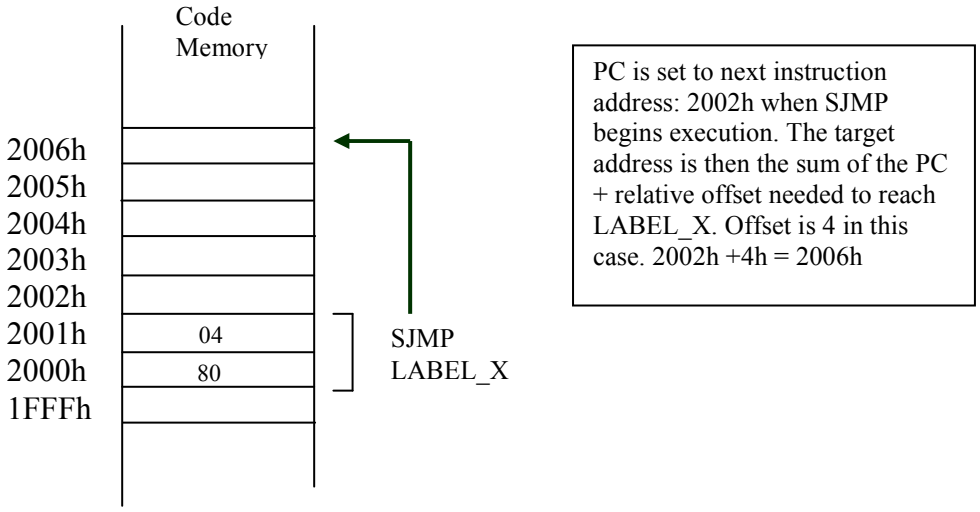
Note, only R0 or R1 can be used as register data pointers for indirect addressing when using MOV instructions.

The 8052 (as opposed to the 8051) has an additional 128 bytes of internal RAM. These 128 bytes of RAM can be accessed only using indirect addressing.

Relative Addressing

This is a special addressing mode used with certain jump instructions. The relative address, often referred to as an offset, is an 8-bit signed number, which is automatically added to the PC to make the address of the next instruction. The 8-bit signed offset value gives an address range of + 127 to –128 locations. Consider the following example:

SJMP LABEL_X



An advantage of relative addressing is that the program code is easy to relocate in memory in that the addressing is relative to the position in memory.

Absolute addressing
Absolute addressing within the 8051 is used only by the AJMP (Absolute Jump) and ACALL (Absolute Call) instructions, which will be discussed later.

Long Addressing
The long addressing mode within the 8051 is used with the instructions LJMP and LCALL. The address specifies a full 16 bit destination address so that a jump or a call can be made to a location within a 64KByte code memory space (2^{16} = 64K). An example instruction is:

LJMP 5000h ; full 16 bit address is specified in operand

Indexed Addressing
With indexed addressing a separate register, either the program counter, PC, or the data pointer DTPR, is used as a base address and the accumulator is used as an offset address. The effective address is formed by adding the value from the base address to the value from the offset address. Indexed addressing in the 8051 is used with the JMP or MOVC instructions. Look up tables are easy to implemented with the help of index addressing. Consider the example instruction:

MOVC A, @A+DPTR

MOVC is a move instruction, which moves data from the external code memory space. The address operand in this example is formed by adding the content of the DPTR register to the accumulator value. Here the DPTR value is referred to as the *base address*

and the accumulator value us referred to as the *index address*. An example program using the indexed addressing mode will be shown later.