Instruction Cycle

The necessary steps that a CPU carries out to fetch an instruction and necessary data from the memory and to execute it, constitute an instruction cycle. An instruction cycle consist of a fetch cycle (FC) and execute cycle (EC).

In fetch cycle, a CPU fetches opcode from the memory. The necessary steps which are carried out to fetch an opcode from the memory, constitute a fetch cycle. The necessary steps which are carried out to get data, if any, from memory and to perform the specific operation specified in an instruction, constitute an execute cycle. The total time required to ~~fetch an opcode~~ execute an instruction is given by

$$IC = FC + EC$$

Machine cycle :-

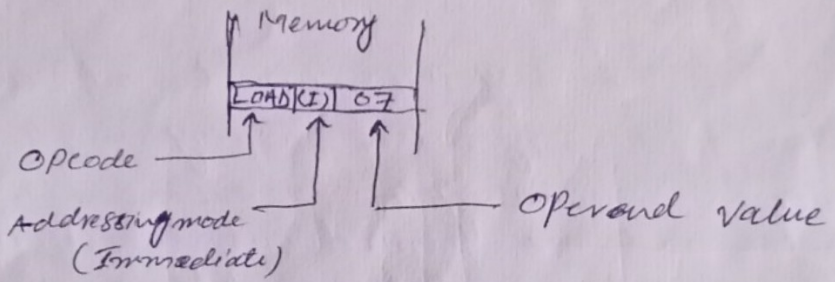
The necessary steps carried out to perform the operation of accessing either memory or I/O device, constitute a machine cycle. In other words, necessary steps ~~involved to read mem~~ carried out to perform a fetch, a read or write operation constitute a "machine cycle".

In a machine cycle one basic operation such as opcode fetch, memory read, memory write, I/O read or I/O write is performed. An instruction cycle consists of several machine cycles. The opcode of an instruction is fetched in the first machine cycle of an instruction cycle. Most of the single-byte instructions require only one machine cycle to fetch the opcode and execute the instruction. Two byte and three-byte instructions require more than one machine cycle. Additional machine cycles are required or needed to read data from or to write data into memory or I/O devices.

Addressing Mode

Each instruction requires certain data on which it has to operate. The techniques to specify data for instructions are called addressing mode. The various addressing modes are

1. Immediate Addressing - In this type of addressing mode the operand is specified within the instruction itself. for example, LOAD IMMEDIATE 7, it is the actual value 7 that is put in the CPU register. In this mode, the operand (data) is in the operand address field of the instruction. Since there is no address field at all, and hence no additional memory accesses are required for executing this instruction. In other words, the actual operand D is A, the content of operand field: i.e D = A. The effective address in this case is not defined.



2. Direct Addressing

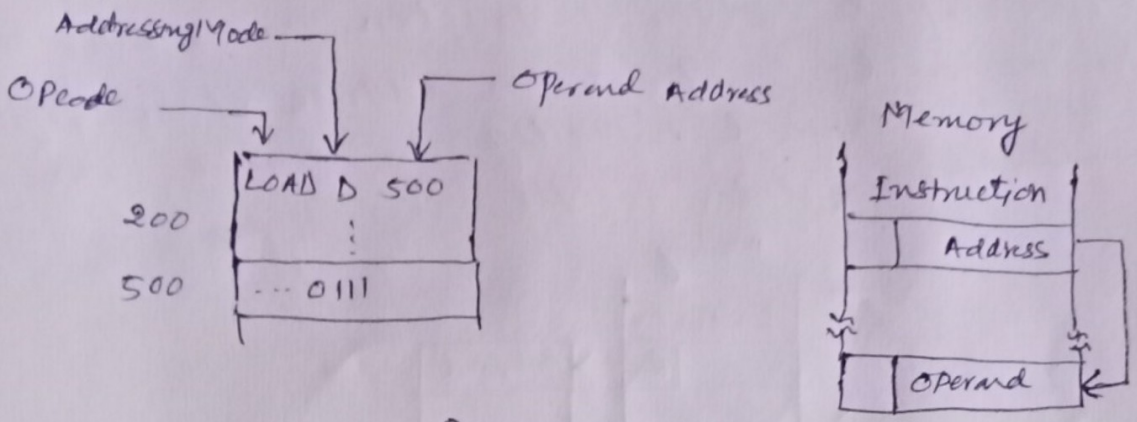
In this scheme the operand field of the instruction specifies the direct address of intended operand. for example, if the instruction LOAD 500 uses direct addressing, then it will result in loading the content of memory cell 500 into the CPU register.

The effective address in this scheme is defined as the address of the operand, that is

~~EA ← A (EA in the example~~

EA ← A (EA in the example will be 500)

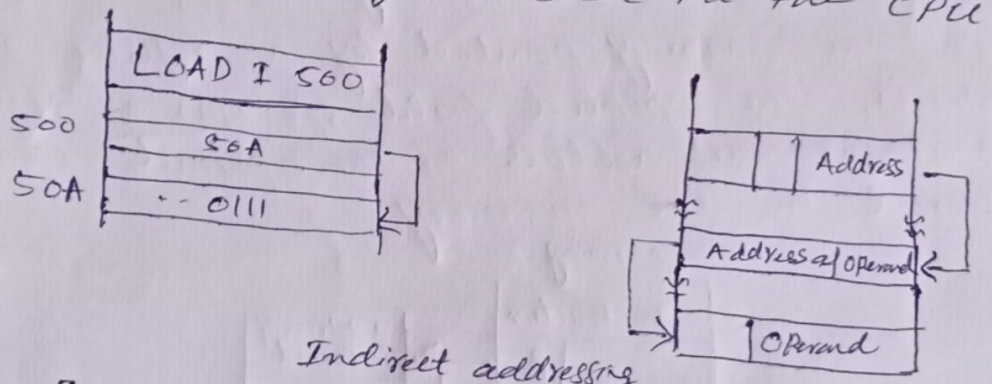
D ← (EA) (D in this example will be 7)



Direct addressing.

3. Indirect Addressing:

In this scheme, the operand field of the instruction specifies the address of the address of intended operand, e.g., if the instruction LOAD I 500 uses indirect addressing scheme, and memory location 500 contents a value of 50A, and memory location 50A contain 7, then value 7 will get loaded in the CPU register.



Indirect addressing

In this addressing scheem, the effective address EA and the content of operand field are related as:

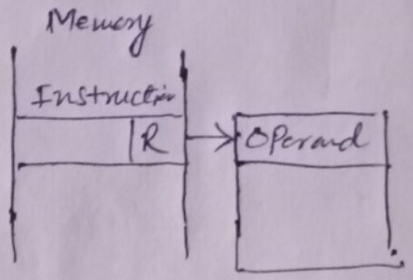
EA = (A) (Content of location 500 i.e 50A)

D = (EA) (Content of location 50A i.e 7)

(4) Register Addressing:

When operands are taken from register, implicitly or explicitly, it is called register addressing. These operands are called register operands. If the operands are from memory location, they are called memory operands. In this scheme, a register address

is specified in the instruction. The register contains the operand. Sometimes the address of the register may be assumed implicitly, for example, the accumulator register.



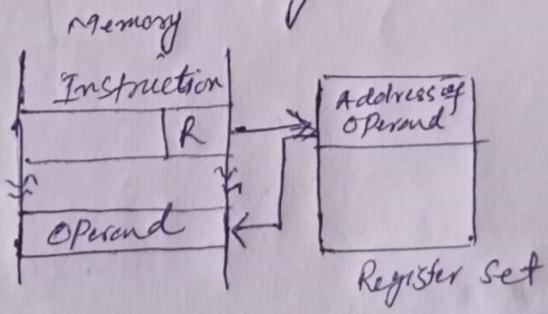
Register addressing.

In this addressing scheme, the effective address is calculated as

$$EA = R$$
$$D = (EA)$$

⑤ Register Indirect Addressing:-

In this addressing scheme, the operand is in the memory pointed by a register. In other words, the operand field specifies a register that contains the address of the operand that is stored in the memory.



Register Indirect addressing

The effective address of the operand in this scheme is calculated as

$$EA = (R)$$
$$D = (EA)$$