Computer System Architecture
COMP201Th
Unit: 2
Basic Computer Organization and Design

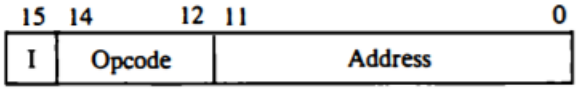
Lecture: 2

Instruction Set

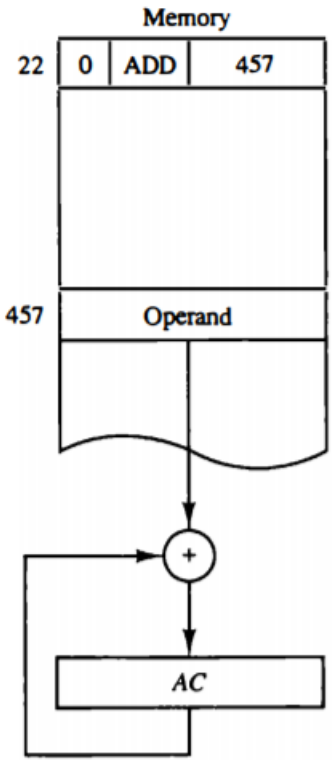
Direct Addressing and Indirect Addressing:

In direct addressing mode, address field in the instruction contains the effective address of the operand and no intermediate memory access is required.

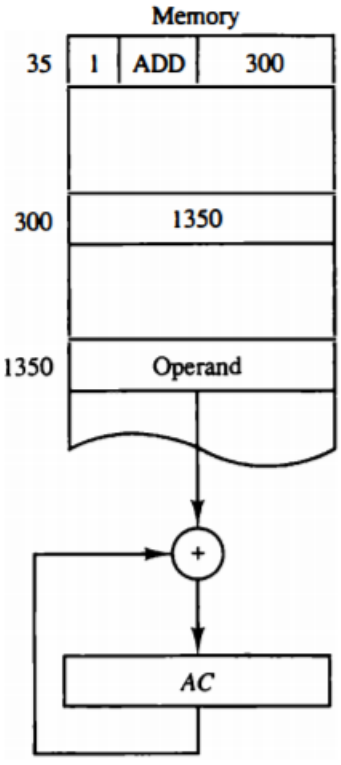
In Indirect addressing mode, address field in the instructions contains the memory location or register where effective address of operand is present. It requires two memory accesses.



(a) Instruction format



(b) Direct address



(c) Indirect address

Computer Instructions:

The basic computer has three instruction code formats as shown below:

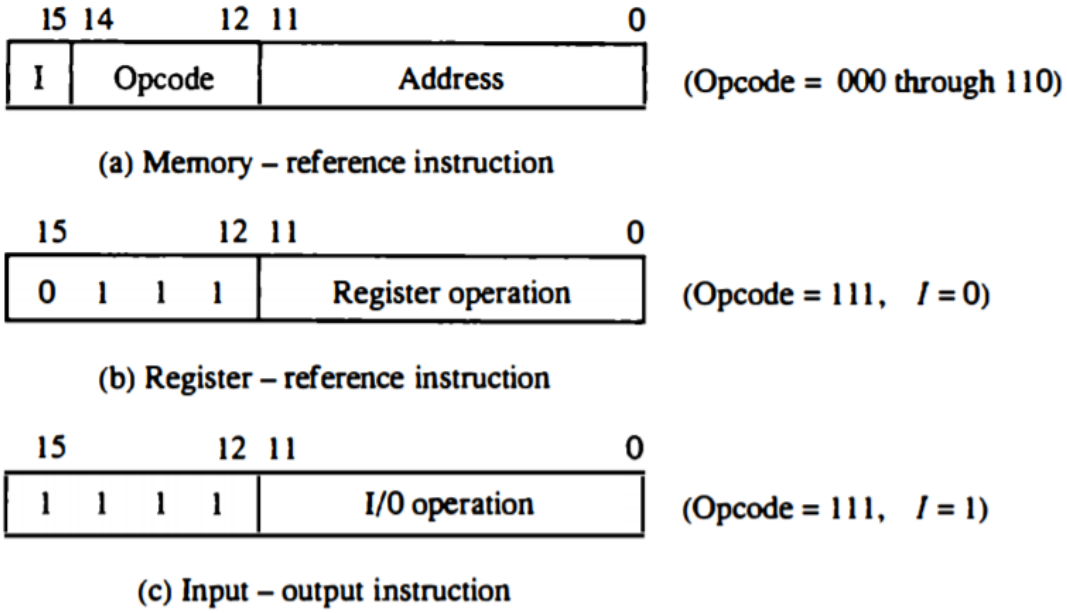


Fig: Basic Computer instruction formats

- **Memory-reference instruction:**
  - A memory-reference instruction uses 12 bits to specify an address and one bit to specify the addressing mode I.
    - I is equal to 0 for direct address and 1 for indirect address.
- **Register-reference instruction:**
  - The register-reference instructions are recognized by the operation code 111 with a 0 in the leftmost bit (bit 15) of the instruction.
  - A register-reference instruction specifies an operation on or a test of the AC register. An operand from memory is not needed; therefore the other 12 bits are used to specify the operation or test to be executed.
- **Input-output Instruction:**
  - An input-output instruction is recognized by the operation code 111 with 1 in the leftmost bit of the instruction.
  - The remaining 12 bits are used to specify the type of input-output operation or test performed.

Total number of instructions chosen for the basic computer is equal to 25 as listed in table below:

Symbol	Hexadecimal code $I = 0$	Hexadecimal code $I = 1$	Description
AND	0xxx	8xxx	AND memory word to *AC*
ADD	1xxx	9xxx	Add memory word to *AC*
LDA	2xxx	Axxx	Load memory word to *AC*
STA	3xxx	Bxxx	Store content of *AC* in memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear *AC*
CLE	7400		Clear *E*
CMA	7200		Complement *AC*
CME	7100		Complement *E*
CIR	7080		Circulate right *AC* and *E*
CIL	7040		Circulate left *AC* and *E*
INC	7020		Increment *AC*
SPA	7010		Skip next instruction if *AC* positive
SNA	7008		Skip next instruction if *AC* negative
SZA	7004		Skip next instruction if *AC* zero
SZE	7002		Skip next instruction if *E* is 0
HLT	7001		Halt computer
INP	F800		Input character to *AC*
OUT	F400		Output character from *AC*
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrupt on
IOF	F040		Interrupt off

Table (i)

Hexadecimal equivalent are used to reduce the 16 bits of an instruction code to four digits with each hexadecimal digit being equivalent to four bits.

- A memory-reference instruction has an address part of 12 bits. The address part is denoted by three x's and stand for the three hexadecimal digits corresponding to the 12-bit address.
    - The last bit of the instruction is designated by the symbol I.
        - When I=0, the last four bits of an instruction have a hexadecimal digit equivalent from 0 to 6 since the last bit is 0.

        - When I=1, the hexadecimal digit equivalent of the last four bits of the instruction ranges from 8 to E since the last bit is 1.
- Register-reference instructions use 16 bits to specify an operation. The leftmost four bits are always 0111, which is equivalent to hexadecimal 7. The other three hexadecimal digits give the binary equivalent of the remaining 12 bits.
- The input-output instructions also use all 16 bits to specify an operation. The last four bits are always 1111, equivalent to hexadecimal F.

Instruction Set Completeness:

A computer should have a set of instructions so that the user can construct machine language programs to evaluate any function. Instruction set completeness refers to the type of instructions that must be included in a computer.

The set of instructions are said to be complete if the computer includes a sufficient number of instructions in each of the following categories:

- **Arithmetic, logical and shift instructions**
    - Arithmetic, logical and shift instructions provide computational capabilities for processing the type of data that the user may wish to employ.
- **Data instructions (for moving information to and from memory and processor registers)**
    - The bulk of the binary information in a digital computer is stored in memory, but all computations are done in processor registers. Therefore, the user must have the capability of moving information between these two units.
- **Program control or Branch**
    - Decision making capabilities are an important aspect of digital computers. E.g. two numbers can be compared and if the first is greater than the second, it may be necessary to proceed differently than if the second is greater than the first. Program control instructions such as branch

instructions are used to change the sequence in which the program is executed.

- **Input and output instructions**
    - Input and output instructions are needed for communication between the computer and the user. Programs and data must be transferred into memory and results of computations must be transferred back to the user.

The instructions listed in Table (i) constitute a minimum set that provides all the capabilities mentioned above.

- There is one arithmetic instruction, ADD, and two related instructions, complement AC(CMA) and increment AC(INC). With these three instructions we can add and subtract binary numbers when negative numbers are in signed-2's complement representation.
- The circulate instructions, CIR and CIL; can be used for arithmetic shifts as well as any other type of shifts desired.
- There are three logic operations: AND, complement AC (CMA), and clear AC(CLA). The AND and complement provide a NAND operation.
- Moving information from memory to AC is accomplished with the load AC (LDA) instruction. Storing information from AC into memory is done with the store AC (STA) instruction.
- The branch instructions BUN, BSA, and ISZ, together with the four skip instructions, provide capabilities for program control and checking of status conditions.
- The input (INP) and output (OUT) instructions cause information to be transferred between the computer and external devices.