Programming the Basic Computer
Lecture: 3
Instruction Code and Machine Language

Instruction Code:
A computer instruction is a binary code that specifies a sequence of microoperations for the computer.

Instruction codes together with data are stored in memory. The computer reads each instruction from memory and places it in a control register. The control then interprets the binary code of the instruction and proceeds to execute it by issuing a sequence of microoperations. Every computer has its own unique instruction set.

The ability to store and execute instructions, the stored program concept is the most important property of a general-purpose computer.

An instruction code is a group of bits that instruct the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part.

The operation code of an instruction is a group of bits that define such operations as add, subtract, multiply, shift and complement. The number of bits required for the operation code of an instruction depends on the total number of operations available in the computer.

The operation code must consist of at least n bits for a given 2^n (or less) distinct operations.

e.g. consider a computer with 64 distinct operations, one of them being an ADD operation. The operation code consists of six bits, with a bit configuration 110010 assigned to ADD operation. When this operation code is decoded in the control unit, the computer issues control signals to read an operand from memory and add the operand to a processor register.

The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory.

An instruction code must therefore specify not only the operation but also the registers or the memory words where the operands are to be found as well as the register or memory word where the result is to be stored.

There are many variations for arranging the binary code of instructions and each computer has its own particular instruction code format. Instruction code formats are conceived by computer designers who specify the architecture of the computer.

How computers are programmed?

Machine Language:

There are various programming languages a user can use to write programs for a computer. However, computer can execute only programs that are represented internally in a valid binary form.

Writing a program for a computer consists of specifying directly or indirectly, a sequence of machine instructions (which is in binary pattern).

Machine instructions are in binary pattern which may be quite difficult to understand and write. So, it is more preferable to write programs in more familiar symbols of the alphanumeric character set.

For a basic computer we have discussed 25 instructions.

- There is three-letter symbol for instructions to make people know what it means.
- The first seven instructions are memory-reference instruction and the others are register-reference and I/O reference instruction.
- The letter M refers to the memory word (operand) found at the effective address.
- The first digit of a register-reference instruction is always 7.
- The first digit of an I/O instruction is always F.

Instruction Set of the *Basic Computer*

Symbol	*Hexa code*	*Description*
AND	0 or 8	AND M to AC
ADD	1 or 9	Add M to AC, carry to E
LDA	2 or A	Load AC from M
STA	3 or B	Store AC in M
BUN	4 or C	Branch unconditionally to m
BSA	5 or D	Save return address in m and branch to m+1
ISZ	6 or E	Increment M and skip if zero
CLA	7800	Clear AC
CLE	7400	Clear E
CMA	7200	Complement AC
CME	7100	Complement E
CIR	7080	Circulate right E and AC
CIL	7040	Circulate left E and AC
INC	7020	Increment AC, carry to E
SPA	7010	Skip if AC is positive
SNA	7008	Skip if AC is negative
SZA	7004	Skip if AC is zero
SZE	7002	Skip if E is zero
HLT	7001	Halt computer
INP	F800	Input information and clear flag
OUT	F400	Output information and clear flag
SKI	F200	Skip if input flag is on
SKO	F100	Skip if output flag is on
ION	F080	Turn interrupt on
IOF	F040	Turn interrupt off

m: effective address
M: memory word (operand) found at m

Machine Language:

Program written for a computer may be in one of the following categories:

i) Binary code: This is a sequence of instructions and operands in binary that list the exact representation of instructions as they appear in computer memory.

ii) Octal or hexadecimal code: This is an equivalent translation of the binary code to octal or hexadecimal representation.

iii) Symbolic code: The user employs symbols (letters, numerals, or special characters) for the operation part, the address part, and other parts of the instruction code. Each symbolic instruction can be translated into one binary coded instruction. This translation is done by a special program called an assembler. Because an assembler translates the symbols, this type of symbolic program is referred to as an assembly language program.

iv)High level programming languages: These are special languages developed to reflect the procedures used in the solution of a problem rather than be concerned with the computer hardware behavior.

An example of a high level programming language is C. It employs problem oriented symbols and formats. The program is written in a sequence of statements in a form that people prefer to think in when solving a problem. However, each statement must be translated into a sequence of binary instructions before the program can be executed in a computer.

The program that translates a high level language program to binary is called a compiler.

Example: Consider a C program to Add two numbers:

```
main ()

{

int a,b,c;

a = 83;

b= -23;

c= a+b;

}
```

• **Binary Program to Add Two Numbers**

Location	Instruction Code
0	0010 0000 0000 0100
1	0001 0000 0000 0101
10	0011 0000 0000 0110
11	0111 0000 0000 0001
100	0000 0000 0101 0011
101	1111 1111 1110 1001
110	0000 0000 0000 0000

• **Hexa program**

Location	Instruction
000	2004
001	1005
002	3006
003	7001
004	0053
005	FFE9
006	0000

• **Program with Symbolic OP-Code**

Location	Instruction		Comments
000	LDA	004	Load 1st operand into AC
001	ADD	005	Add 2nd operand to AC
002	STA	006	Store sum in location 006
003	HLT		Halt computer
004	0053		1st operand
005	FFE9		2nd operand (negative)
006	0000		Store sum here

• **Assembly-Language Program**

```
      ORG   0     /Origin of program is location 0
      LDA   A     /Load  operand from location A
      ADD   B     /Add  operand from location B
      STA   C     /Store sum in location C
      HLT         /Halt computer
A,    DEC   83    /Decimal operand
B,    DEC   -23   /Decimal operand
C,    DEC   0     /Sum stored in location C
      END         /End of symbolic program
```

A machine language program is a binary program of category 1 (i.e. binary code) and as there is simple equivalency between binary and octal or hexadecimal representation, category 2 is also referred as machine language.