MCA Part I

Paper-II

Topic: Instruction Set Architecture

Prepared by: Dr. Kiran Pandey

Email-id: kiranpandey.nou@gmail.com

Introduction

The internal organization of a digital system is defined by the registers it employs and the sequence of micro-operations it performs on data stated in the registers. A digital computer is a general purpose digital system capable of executing various operations and in addition can be instructed as to what specific sequence of operations it must perform . The user of a computer can control the process by means of programs that is a set of instruction that specify the operations, operands and the sequence in which processing has to occur.

The Instruction Set Architecture (ISA) is the part of the processor that is visible to the programmer or compiler designer. They are the parts of a processor design that need to be understood in order to write assembly language, such as the machine language instructions and registers. The ISA serves as the boundary between software and hardware.

Instruction Set Characteristics

The CPU is the heart and brain of the computer .The entire processing takes place in the Central Processing Unit (CPU). It performs calculations, issues the commands, coordinates with all other hardware components, and executes programs including the operating system. But to make CPU work, we must speak to it in binary machine language. In other words it is group of bits that tells the computer to

perform specific operations. The collection of bits of a machine language are known as instructions, and its syntax is known as an instructions set.

Instruction set is the boundary where the computer designer and computer programmer see the same computer from different viewpoints. From the designer, point of view, the computer instruction set provides a functional description of a processor, that is :

(i) A detailed list of the instructions that a processor is capable of processing.

(ii) A describes of the types/locations/access methods for operands.

The common goal of computer designer is to build the hardware for implementing the machine's instructions for CPU. From the programmer's point of view, the user must understand machine or assembly language for low-level programming. Moreover, the user must be aware of the register set, instruction types and the function that each instruction performs. However, our prime focus is the programmer's viewpoint with the design of instruction set.

Instruction set is the collection of machine language instructions that a particular processor understands and executes. In other words, a set of assembly language mnemonics represents the machine code of a particular computer. Therefore, if we define all the instructions of a computer, we can say we have defined the instruction set. it should be noted here that the instructions available in a computer are machine dependent, that is, a different processors have different instruction sets. However, a newer processor that may belong to some family may have a compatible but extended instruction set of an old processor of that family. Instructions can have different formats. the instruction format includes the following components:

- the instruction length;
- the type;
- length and position of operation codes in an instruction; and
- the number and length of operand addresses etc.

Each instructions consists of several fields. The most common fields found in instruction formats are:

Opcode : It specifies the operation (ADD, SUBTRACT, MULTIPLY, etc.) to be performed.

Operands : An address field of operand on which data processing is to be performed.

- An operand can reside in the memory or a processor register or can be incorporated within the operand field of instructions as an **immediate constant**. Therefore a mode field is needed that specifies the way the operand or its address is to be determined.

A sample instruction format is given in figure 1.

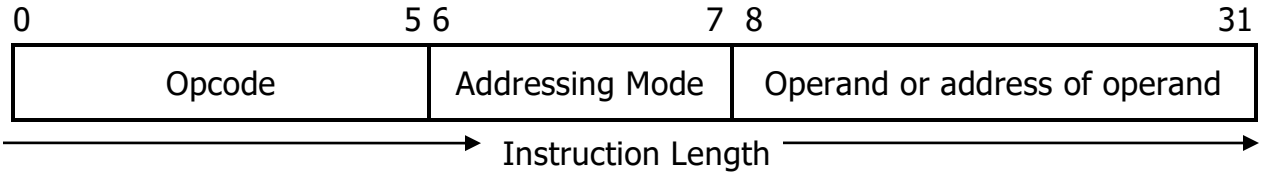

Figure 1 : An Instruction Format of 32 bits

From the above figure we have the following observations :

(i) The size of the opcode is 6 bits. So we will have $2^6 = 32$ operations.

(ii) There is only one operand address machine.

(iii) There are two bits for addressing modes. Therefore , there are $2^2 = 4$ different addressing modes possible for this machine.

(iv) The last field (8 - 31 bits = 24 bits) here is the operand or the address of operand field.

In case of immediate operand the maximum size of the unsigned operand would be 2^{24}. In case it is an address of operand in memory, then the maximum physical memory size supported by this machine is 2^{24} = 16 MB.

The opcode field of an instruction is a group of bits that define various processor operations such as LOAD, STORE, ADD, and SHIFT to be performed on some data stored in registers or memory. The operand address field can be data, or can refer to data - that is address of data, or can be labels, which may be the address of an instruction we want to execute next. Such labels are commonly used in Subroutine call instructions. An operand address can be:

- The memory address
- CPU register address
- I/O device address

The mode field of an instruction specifies a variety of alternatives for referring to operands using the given address. **It is important to know that if the operands are placed in processor registers then an instruction executes faster than that of operands placed in memory, as the registers are very highspeed memory used by the CPU. However, to put the value of a memory operand to a register we will require a register LOAD instruction.**

Instruction is represented as a sequence of bits. A layout of an instruction is termed as instruction format. Instruction formats are primarily machine dependent. A CPU instruction set can use many instruction formats at a time. Even the length of opcode varies in the same processor.

A computer can have a large number of instructions and addressing modes. The older computers with the growth of integrated circuit technology have a very large and complex set of instructions. These are called "Complex Instruction Set Computer" (CISC). Examples of CISC architectures are the Digital Equipment Corporation VAX computer and the IBM 370 computer. But it was found later that many complex instructions found in CISC are not used by the program. This lead to the idea of making a simple but faster computer, which could execute simple instructions much faster. These computers have simple instructions, registers addressing and move registers. These are called Reduced Instruction Set Computers (RISC). We will study more about RISC in unit 4 of this Block.

Instruction Set Design Considerations

Some of the basic considerations for instruction set design include selection of:

- A set of data types (e.g. integers, long integers, doubles, character strings etc.).
- A set of operations on those data types.
- A set of instruction formats. Includes issues like number of addresses, instruction length etc.
- A set of techniques for addressing data in memory or in registers.
- The number of registers which can be referenced by an instruction and how they are used.

Operand Data Types

Operand is that part of an instruction that specifies the address of the source or result, or the data itself on which the processor is to operate. Operand types usually give operand size implicitly. In general, operand data types can be divide in the following categories.

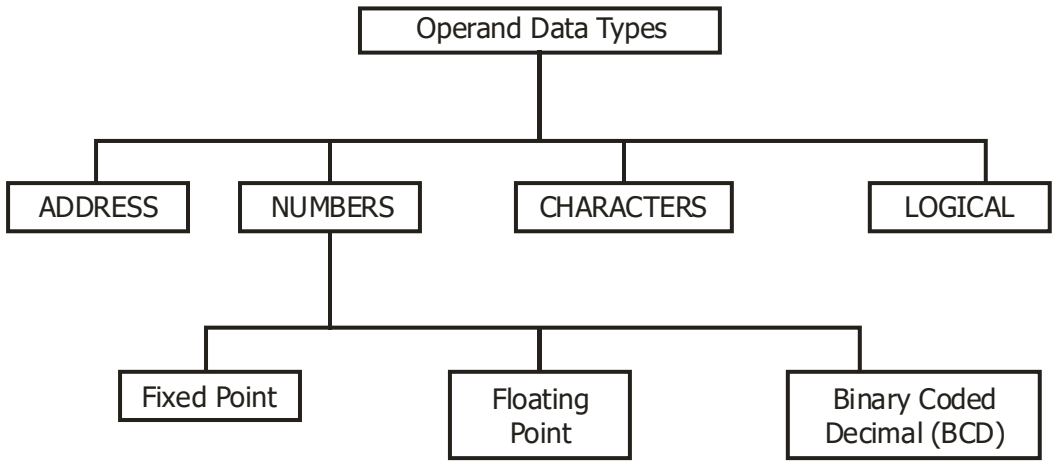

Figure 2 : Operand Data Types

- **Addresses :** Operands residing in memory are specified by their memory address and operands residing in registers are specified by a register address. Addresses provided in the instruction are operand references.
- **Numbers :** All machine languages include numeric data types. Numeric data usually use one of three representations:
- **Floating-point numbers-single precision** (1 sign bit, 8 exponent bits, 23 mantissa bits) and double precison (1 sign bit, 11 exponent bits, 52 mantissa bits).
- **Fixed point numbers (signed or unsigned).**
- **Binary Coded Decimal Numbers(BCD).**
- **Characters:** A common form of data is text or character strings. Characters are represented in numeric form, mostly in ASCII (American Standard Code for Information Exchange). Another code used to encode characters is the Extended Binary Coded Decimal Interchange Code(EBCDIC).
- **Logical data:** Each word or byte is treated as a single unit of data. When an n-bit data unit is considered as consisting of n 1-bit items of data with each item having the value 0 or 1, then they are viewed as logical data. Such bit-oriented data can be used to store an array of Boolean or binary data variables where each variables can take on only the values 1 (true) and 0 (false). One simple application of such a data may be the cases where we manipulate bits of a data item. For example, in floating-point addition we need to shift mantissa bits.