An Overview of Computers and Microprocessors

1

Objectives

At the conclusion of this chapter you should be able to:

1. *Understand overview of computer organization and its various units.*
2. *Note the generations of computers with respect to their hardware and software.*
3. *Know the advances in computer architecture.*
4. *Realize the advances in electronic integrated circuit technology.*
5. *Appreciate evolution of microprocessors due to Moore's law.*

We live in a computing and computer oriented society, and we constantly come across multitude terms relating to computer organization, architecture, and circuits. Before getting started with main flow of the book, we will understand these basics and how they are interlinked to microprocessors.

COMPUTERS: AN OVERVIEW

Computing is intimately tied to the representation of decimal numbers. Advances in decimal number systems and mathematical notations eventually lead to the formulation of mathematical operations such as addition, subtraction, multiplication, division, square root, etc. Earlier, these mathematical operations were done using human fingers, thereafter by using pencil and paper or chalk and slate. The concept of programmable computers was developed by an English mechanical engineer Charles Babbage in around 1820. Programmable computer implies a machine which can do mathematical operations by some components called *hardware*, and the mathematical operations and the sequence in which they are worked out is controlled by a 'program' referred as *software*. The hardware consists of active and passive interconnected components and the software consists of commands which dictate the hardware to do mathematical operations on the data in a desired sequence. Passive components will consume energy and are incapable of producing energy such as resistors, capacitors and inductors, while active components produce energy or power gain. With the advances in electronics, the fist active component was developed in the year 1906 which was called *triode*, a three-electrode vacuum tube. Higher versions of this vacuum tube were invented later. Using this active component and other passive components, the first digital computer was built in 1946. Even though most people are familiar with the decimal number system, an electronic device with an active component cannot be reliably built with 10 distinguishable states. Therefore, computers use the binary number system which has only two states, 0 and 1. An active device can be made fully conducting (representing a state) and can be made not to conduct at all, representing the other state of a binary system. These two states can be reliably built with an active device. After vacuum tubes, the transistor—a semiconductor device—was invented and computers were built with transistors. Further improvements in semiconductor technology resulted in placing many transistors on a silicon die. These developments resulted in active devices in small-scale integration, medium-scale integration, very large sacle integration, and so on. Computers were built with these devices resulting in generations of computers. A computer will have a CPU or central processing unit, a main memory unit to store information, and input and output devices to feed in information and printing out information. Table 1.1 shows the generations of computers, what devices were used and what the hardware was made of to do computations under the control of various levels of software. Like advancements in active devices, there are also advances in memory devices, input and output devices and software. Main memory devices started with relays, moved to magnetic core memories and now there are highly advanced, high volume and very fast dual-data-rate dynamic memories in VLSI technology. Input and output devices started with paper tapes, card punch to the present keyboards and then moved on to monitors, hard disks and high-speed printers, scanners, etc. Secondary memories are initially magnetic drums, magnetic tapes and now high-speed magnetic disks, optical disks, and so on. Software too started

Table 1.1 Generations of Computers

Generation	*Timeline*	*Technology and Architecture*	*Software and Applications*	*Systems*
First	(1945–54)	Vacuum tubes Relay memories CPU driven by PC and accumulator Fixed-point arithmetic	Machine and assembly language Single-user basic I/O using programmed and interrupt mode	ENIAC, TIFRAC, IBM 701, Princeton IAS
Second	(1955–64)	Discrete transistors Core memories Floating point; Arithmetic I/O Processors Multiplexed memory access	HLL used with compilers Batch processing Monitoring Libraries	IBM7099, CDC 1604, MINSK-II
Third	(1965–74)	Integrated circuits Microprogramming Pipelining Caching Lookahead processing	Multiprogramming Time sharing OS Multi-user applications	IBM 360/700, CDC 6000, TA-ASC, PDP-8, EC-1030
Fourth	(1975–84)	LSI/VLSI and semiconductor memory Microprocessor technology Multiprocessors Vector super-computing Multicomputer	Multiprocessor OS Languages Compilers	VAX 9800, Cray X-MP, IBM 3600, Pentium Processor based systems (PCs), UltraSPARC, etc.
Fifth	(1991–present)	VSLI/VHSIC processors Scalable architecture	Massively parallel processing Grand challenge applications	Cray/MPP, TMC/CM-5, Intel paragon, Fujitsu VP500

with machine language, assembly language and moved on to higher level languages such as FORTRAN, COBOL, C, C++, JAVA, etc., which run under an operating system. The operating system, also known as an OS, is a software that communicates with computer hardware at the most elementary level. Without an operating system, no software programs can run. The OS is what allocates memory, processes tasks, accesses disks and peripherials, and serves as the user interface.

INFORMATION REPRESENTATION IN COMPUTER SYSTEMS

In a computer, information is represented in a binary form for reliability. The information could either be machine instructions or data. Machine instructions and data representation is discussed in the following sections. The input which is either in Assembly or Higher Level Language (HLL) is converted into standard 7 or 8 binary characters, called ASCII-7 and ASCII-8 (American Standard Code for Information Interchange) or EBCDIC (Extended Binary Coded Decimal Interchange Code) and sent to the CPU. The CPU sends this information to the memory. The input can also be sent directly to the memory using a device called Direct Memory Access (DMA). DMA will be discussed in later chapters. The information sent from an input device could be either machine instructions or data. The binary-coded information sent by the input device will be converted to pure binary form before the processor starts processing.

CPU fetches the machine instructions from the main memory and executes them on the data from the main memory, thereby altering the data as dictated by machine instruction.

Instruction Formats and Addressing Modes

Machine instruction consists of an *opcode* field telling the computer what function it has to perform on the *operand* (*data*) specified in another field of the machine instructions. The opcode could be of fixed length or of variable length. Data could be specified in the machine instruction itself or its address in main memory could be given. There are instructions with one, two or three operand addresses. In single-address machine instruction, data is fetched from the memory and the function specified by the opcode is performed on this data and data in another register in the CPU, and the result is left in the register. This register is usually called the *accumulator*. In two-address machines, data is fetched from the two addresses specified in the instruction and the operation as specified by the opcode is done on the data. The result is stored in the memory in one of the addresses specified in the instruction. In a three-address machine, data is fetched from the two addresses given in the instruction, the operation performed and the result stored in the third address. Mostly, either single-address or two-address machine instruction formats are used. To give programming flexibility, the operand addresses have addressing modes described below.

REGISTER ADDRESSING MODE

The address specified is not the main memory address but the address of one of the internal registers of the CPU.

DIRECT ADDRESSING MODE

The operand address specified points to the operand in the main memory.

INDIRECT ADDRESSING MODE

The address specified is not the operand address. This address points to the memory location where the address of the operand is available.

BASE ADDRESSING

The address part in the instruction will be added to the contents of a register in CPU called *base register* and this becomes the effective address of the operand.

INDEX ADDRESSING

The contents of the address part specified in the instruction will be added to a register in the CPU called the *index register* and this becomes the effective address of the operand.

IMMEDIATE ADDRESSING

The operand itself is in the instruction.

IMPLIED ADDRESSING

No operand address is given in the instruction. The opcode itself specifies the register in the CPU where the operand is available.

RELATIVE ADDRESSING

The address part in the instruction is added to the contents of the Program Counter (PC). This beçomes the effective address of the operand.

Data Formats

In a computer, the data is represented in fixed-point, floating-point and decimal formats.

FIXED POINT

The data is represented in binary form and the binary point is assumed to be on the right side of the last bit which is called the Least Significant Bit (LSB). Numbers could be signed or unsigned. In an unsigned binary number, the bit at the leftmost position is called the Most Significant Bit (MSB). In a signed number, the most significant bit is the sign bit. '0' is used for positive numbers and '1' is used to represent negative numbers. Signed numbers can be represented either in signed magnitude, or signed one's complement, or signed two's complement form. In *signed magnitude form*, the magnitude of the number is in pure binary form for both positive and negative numbers. In *signed 1's complement form*, for positive numbers, the magnitude is in pure binary form and for negative numbers, the magnitude is not in pure binary form but is in 1's complement form. In 2's complement representation, positive numbers have the magnitude in pure binary form but for negative numbers, the magnitude is in 2's complement form. Computers were built in all the three representations but most computers and computers of today use 2's complement representation for fixed-point numbers. In fixed point, all numbers are in integer form.

To handle a wide range of numbers, we can resort to floating-point number representation.

Fixed-point arithmetic is exact whereas floating-point arithmetic is not exact and will not give accurate result. This is to be borne in mind.

You may have to differentiate between two terms: accuracy and precision.

ACCURACY vs. PRECISION

Accuracy The quality of freedom from mistake or error, of conformity to truth or to a rule.

Precision The degree of exactness or discrimination with which a quantity is stated.

Accuracy is distinguished from precision in the following example: A six-place table is more precise than a four-place table. However, if there are errors in a six-place table, it may be more or less accurate than the four-place table (IEEE Standard Dictionary).

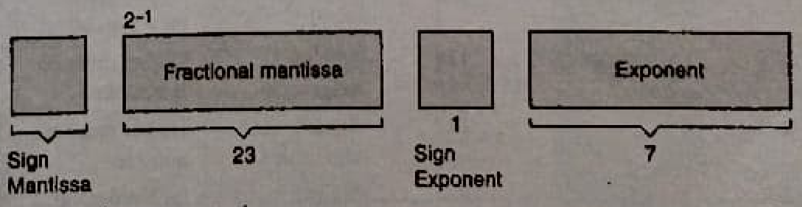

Fig. 1.1 Single-precision floating-point representation.

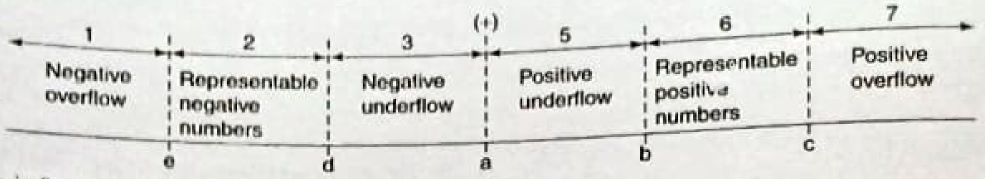

Fig. 1.2 Overflows in floating-point representation.
NOTE: Numbers in regions 2, 4, 6 are representable.

FLOATING-POINT REPRESENTATION

In scientific computing, we need:

1. **To be able to represent and perform arithmetic operation on a wide range of numbers**
2. **Even if the number is representable, sometimes it should be very convenient to express a number, e.g., 1000000 as 1.0×10^6.**
3. **Sign of mantissa**
4. **Mantissa**
5. **Sign of exponent**
6. **Exponent**

A 32-bit number is called *single-precision floating point representation* and a 64 bit floating point number is called *double-precision representation*.

The mantissa is in fractional form and the binary point is to left of the most significant bit in most machines. The exponent is an integer.

The maximum number that could be written: $+(1-2^{-23}) \times 2^{+127} = 10^{38}$

Minimum number $= 2^{-23} \times 2^{-127}$

The other representation of floating-point format is shown in Fig. 1.2:

The exponent will not have a sign bit and it is called an *exponent with bias*. For an 8-bit exponent, the bias is 128 or 127. There are two types of floating points:

- **Normalized floating point**
- **Unnormalized floating point**

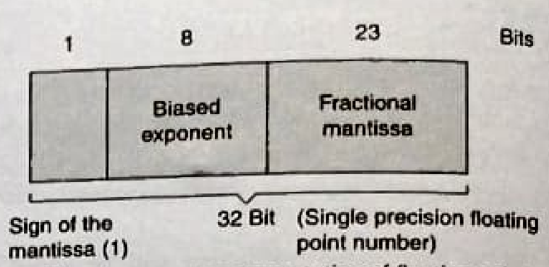

Fig. 1.3 Biased exponent representation of floating-point number.

In normalized floating point, the first bit (MSB) has to be '1'. Most systems, even IEEE format, have normalized floating-point numbers because it maintains higher precision than unnormalized floating point. In the normalized floating-point representation, since the first bit has to be non-zero, to increase the precision of floating-point arithmetic, floating-point numbers in the memory will be written with implicit 1st bit. In other words, the first bit of the mantissa after the binary point will be given a weightage of 2^{-1}. This will obviously increase the length of the mantissa by one more bit. When the implicit first bit floating-point numbers come to the floating-point arithmetic unit, the implicit first bit is made explicit. So in the floating-point arithmetic unit, the registers that hold the numbers have some additional bits. These additional bits are called *guard bits*. It can be seen that two guard bits would be sufficient to maintain the maximum precision of the floating-point arithmetic.

Table 1.2 Floating-point Representations used in Various Machines

Computer	*Radix of Exponent(Base)*	*Bits in Exponent*	*Number System for Exponent in Excess Code*	*Number System for Mantissa*	*A Number is Normalized if*	*Mantissa as a*	*Approx. Largest Number*
IBM 360/370	16	7	64	Signed magnitude	Higher-order digit is non-zero	Fraction	10^{76}
Cyber 70	2	11	1024	1's complement	Higher order bit $\neq$ sign bit	Integer	10^{322}
PDP 70	2	8	128	Signed magnitude	Higher order bit is non-zero	Fraction	10^{38}
MIPS	2	8	127	Signed magnitude	Higher order bit non-zero	fraction	10^{38}
RISC	2	11	1023	Signed magnitude	high order bit nonzero	fraction	10^{323}

The base is 2 in case of most machines including Intel processors, and 16 in case of IBM machines. Mantissa can be in signed magnitude form or signed complement form but most machines use signed-magnitude representation for mantissa. Table 1.2 shows the floating-point representation used in some selected machines.

How to represent a Floating-point Zero In fixed-point representation, zero is represented by all zeros in the number. If the sign bit is '1' and all magnitude bits are zero, it is negative zero but in this case the sign bit is complemented and the representation is fixed-point zero.

In a floating-point number, once the mantissa is '0', the exponent can be anything. If the entire mantissa is '0' and the exponent has any value other than maximum negative exponent, the result is '0' but it is called *dirty zero*. A clean 'zero' is one where you have all 'zeros' in the mantissa and the maximum negative exponent.

DECIMAL REPRESENTATION

Representation of Decimal Digits in ASCII/EBCDIC Formats Each design will have an 8-bit representation. The first four bits represent the 'zone' 1111-EBCDIC and 0101 for ASCII-8, the next four bits represent the decimal number 'in one of the BCD codes but mostly in 8-4-2-1 code.

There are two types of representation: zoned and packed formats. When the decimal information goes through the input and output devices, the format will be zoned and within the computer, it is in packed format. Zoned and packed formats are shown in Fig. 1.4.

Packed Decimal When a digit is sent through the input unit, it stays in ASCII 8 or EBCDIC format.

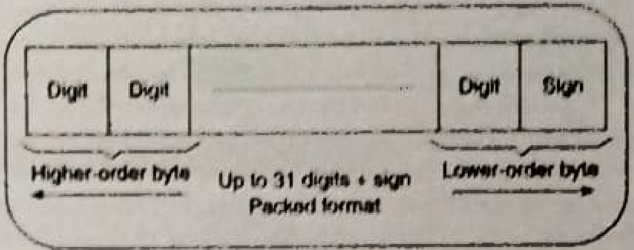

Fig. 1.4 Packed format

Zoned Format

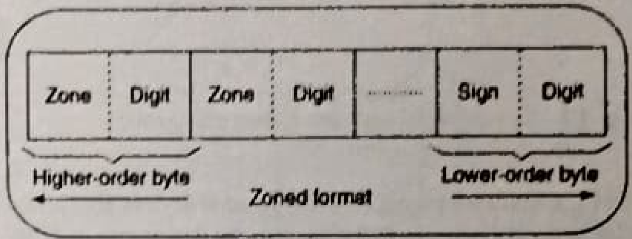

Fig. 1.5 Zoned and packed decimal formats.

ORGANIZATION OF COMPUTER SYSTEMS

A simple computer has a Central Processing Unit (CPU), Main Memory (MM), secondary memory (magnetic disc), and input/output interfaces and devices, all interconnected as shown in Fig. 1.6. The input, is either in assembly or higher level language (HLL), is converted into standard 7 or 8 binary coded characters, called ASCII-7 and ASCII-8 (American Standard Code for Information Interchange) or EBCDIC (Extended Binary Coded Decimal Interchange Code) and sent to CPU. Then the CPU sends this information to the memory. The input can also be sent to the memory directly using a device called Direct Memory Access (DMA) which is not shown in Fig. 1.6.

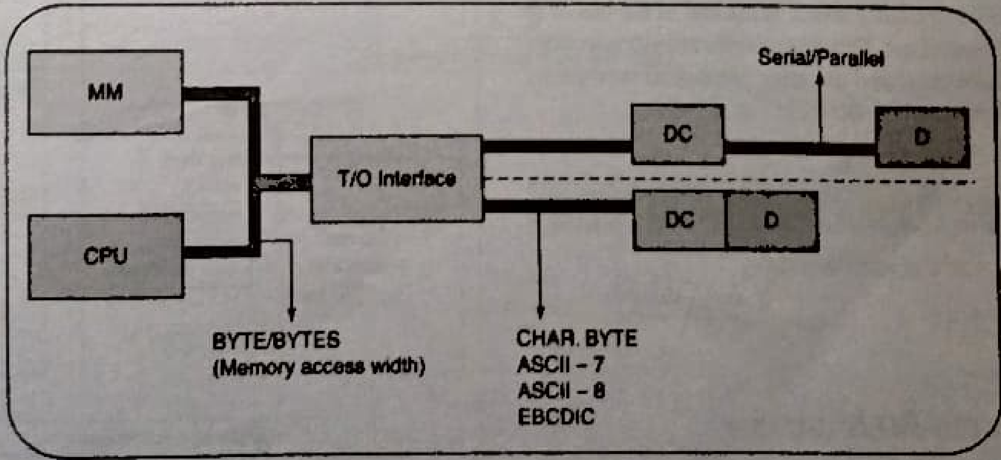

Fig. 1.6 Basic computer system organization.

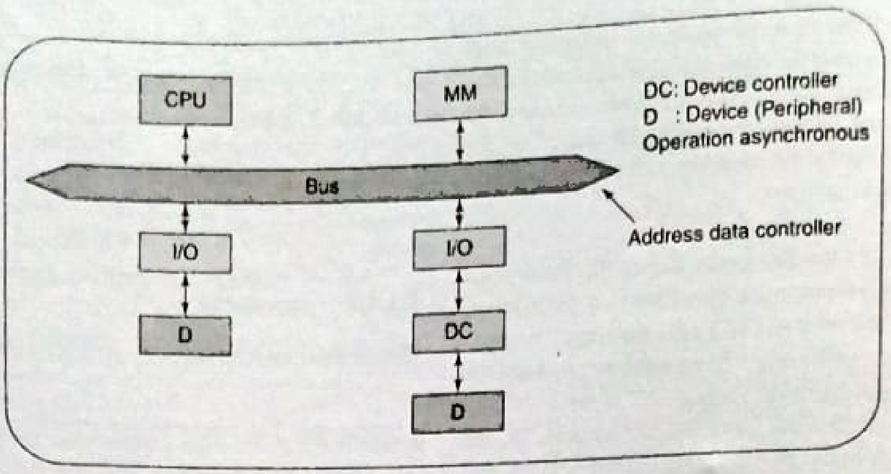

Fig. 1.7 Bus-organized computer system configuration.

DMA is discussed in later chapters. In some devices, the Device Controller (DC) can control more than one device such as a disk and in others, the device controller and device will be integrated together such as printers. The binary coded information (program) which is sent through the input device to the main memory is converted into binary form before the CPU processes it.

The CPU fetches the machine instructions from the main memory and executes them on the data from the main memory, thereby altering the data as dictated by machine instructions.

Connecting various units as shown in Fig. 1.6 consumes lots of wires and it is not very convenient either to add additional units or debug the system in case of any failure. Hence, the concept of bus was introduced. A bus is a conducting wire carrying a signal from a transmitting source and delivering the signal to one or more receivers. There are two types of buses:

- Unidirectional
- Bidirectional

The frequencies at which the signals are transmitted on the bus make it important that the bus be treated as a transmission line that needs to be terminated at both ends for a bidirectional bus with impedances which equals the characteristic impedance of the wire. A unidirectional bus can be terminated with the characteristic impedance either at the end of the bus or in series with the transmitter. This termination avoids reflections on the bus and assures signal integrity. Most of the computers today are bus organized as shown in Fig. 1.7.

ARCHITECTURE OF COMPUTERS

There are two computer architectures:

- von Neumann
- Harvard

von Neumann Architecture

In the von Neumann architecture, there is only one main memory in which both instructions and data are stored together, the process being known as *stored program concept*. When information is fetched from the main memory location address through a register called *program counter*, that information is a machine instruction. If the information is fetched with the address specified from any other register, then that information is data. This *stored-program* concept was developed by designers of ENIAC, a vacuum-tube-based machine built for the US Army between 1943 and 1946. The concept was first expounded by von Neumann (1945), and incorporated into the IAS computer (at the Princeton Institute for Advanced Studies) which was completed in 1952. All general-purpose computers are now based on the key concepts of the von Neumann architecture (Fig. 1.8).

Though the von Neumann model is universal in general-purpose computing, it suffers from one clear problem. All information (instructions and data) must flow back and forth between the processor and memory through a single channel, and this channel has a finite bandwidth. When this bandwidth is fully

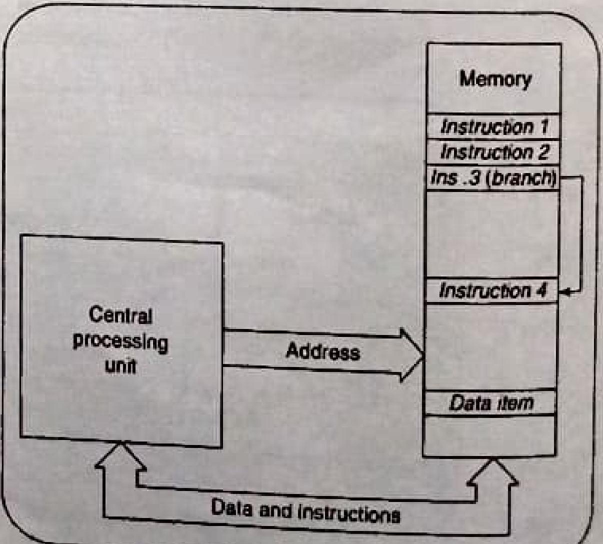

Fig. 1.8 The von Neumann architecture.

used, the processor can go no faster. This performance-limiting factor is called *von Neumann bottleneck.*

Harvard Architecture

The other architecture, known as Harvard architecture, shown in Fig. 1.9, has two memories—one for instructions and the other for data. The name comes from Harvard Mark 1, an electromechanical computer which pre-dates the stored-program concept of von Neumann, as does the architecture in this form. The advantage of this architecture is increased bandwidth available due to separate buses for instructions and data. The disadvantage is that the storage is allocated to instructions and data in a fixed ratio.

In current-day computers, von Neumann architecture is used in principle but within the CPU, there are two memories called *instruction cache* and *data cache*. The main memory will have both instructions and data together. The cache (hidden) memory is a very fast memory of limited size within the CPU. More details about the cache memory is discussed later. The modified Harvard architecture is shown in Fig. 1.10.

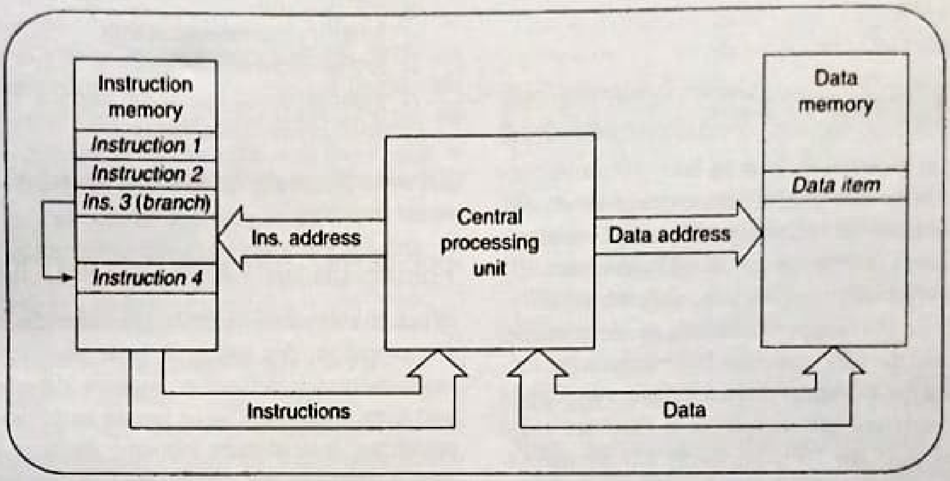

Fig. 1.9 The Harvard architecture.

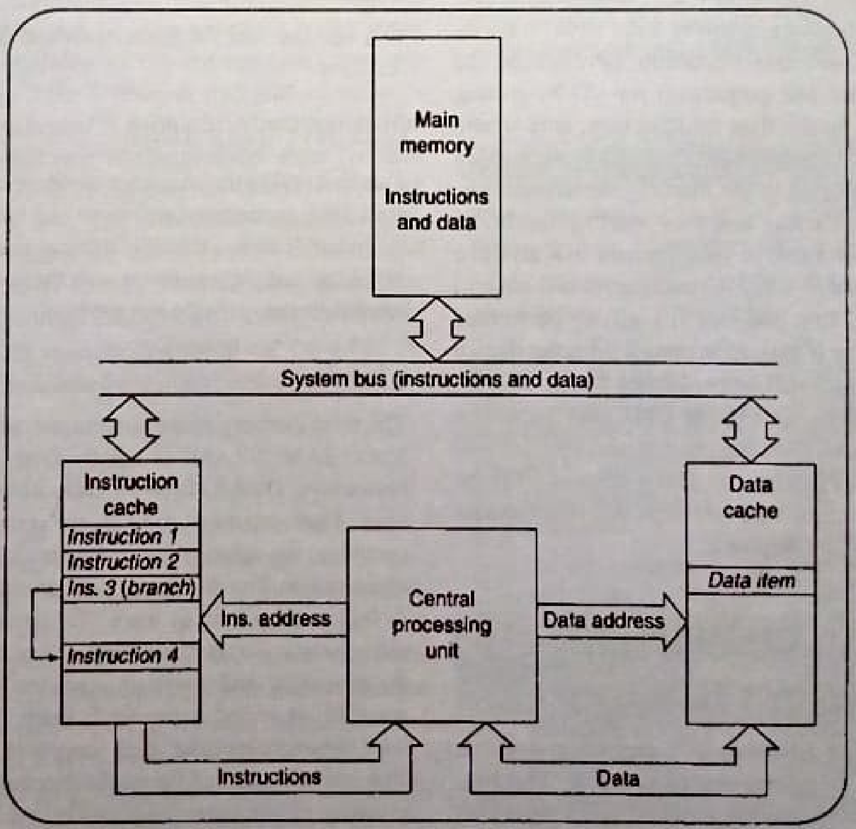

Fig. 1.10 A modified Harvard architecture.

INTERRUPTS AND INTERRUPT PROCESSING

Events occurring asynchronously/synchronously send signals to CPU to inform their occurrence for CPU to take appropriate action. These signals are called *interrupts*. Whenever an interrupt comes and if it is to be serviced, CPU saves the current program it is executing and goes to another program to service the required operation corresponding to the accepted interrupt.

Interrupts are discussed in detail in the chapter on input, output and interfacing.

INPUT/OUTPUT

Input and output in a computer can be done either in programmed I/O mode by the CPU or interrupt mode by the CPU after it receives an interrupt signal from I/O devices, direct memory access (DMA) or by an I/O processor. I/O can be done either in I/O mapped I/O mode or memory-mapped I/O mode. In I/O mapped I/O mode, devices will be given addresses, and the CPU executes I/O machine instructions by specifying the I/O address. In the memory-mapped I/O, I/O device registers are treated as if they are main memory locations, and the CPU performs I/O by executing memory reference instructions by specifying the main memory address assigned to that device. So in the memory-mapped I/O, some small part of the main memory address is allotted to I/O/. I/O operation involves three steps: i) initiation, ii) data transfer, and iii) termination. In initiation, the CPU selects the device and prepares it for I/O by issuing non-data commands. In the data transfer step, data would be transferred between the I/O device and CPU to memory or directly from the device to the memory. In termination, the CPU examines the status and sees whether the I/O is performed without any error. In programmed I/O, all three steps will be performed by CPU. In interrupt mode, step (a) is performed by CPU first and step (b) will be performed by the CPU only when it gets an interrupt from the device for data transfer. Step (c) will be performed by the CPU. In DMA, the step (a) will be done by the CPU, step (b) will be done by DMA without CPU intervention and step (c) will be performed by the CPU when the DMA informs CPU the end of data transfer through an interrupt. All these modes are discussed in detail in Chapter 9.

ADVANCES IN COMPUTER ARCHITECTURE

There have been major advances in computer systems as could be seen in the computer-generation table. The simple CPU had only fixed-point Arithmetic and Logic Unit (ALU), accumulator, program counter, and some registers such as index register, base register, and central control unit. With technological developments, the simple CPU has now become a superscalar CPU which consists of the following units:

a) Bus Interface Unit (BIU)
b) Prefetch unit and instruction queue
c) Decoding unit(s)
d) Instruction Cache (I CACHE)
e) Data Cache (D CACHE)
f) Branch Target Cache (BTC)
g) Control unit
h) Memory management unit
i) Integer operation units
j) Floating-point operation units
k) Special functional units

Let us briefly study the functions of each block of the superscalar processor.

PRE-FETCH UNIT AND INSTRUCTION CACHE

When an instruction is fetched, decoded and sent to the ALU for execution, this unit will fetch other instructions in the main memory which are in sequence, decode the instruction, and keep them ready to be passed on to the execution units which are free, thereby reducing the time for fetching the next instructions and making the CPU run faster.

DECODING UNIT

This unit decodes the instructions and decides the actions to be taken.

BUS INTERFACE UNIT

This unit makes the processor communicate with the external units such as memory, and input and output devices through the buses. It makes the CPU internal signals compatible both in voltage and current levels with the external world and also communicates with the bus protocol.

CACHE

The main memory of the computer system is either based on Static RAM (SRAM) or mostly Dynamic RAM (DRAM). Nowadays, DDRAMs (Dual Data Rate DRAMs) are being used. These memory devices are comparatively slower in providing the information to the processor which runs many times faster. Processor registers provide the fastest access to information held in them. So the need was felt to provide a relatively small amount of very fast memory between the processor and the main memory. This fast memory in the CPU is called a *cache*. It holds copies of repeatedly used instructions and data stored in the main memory. The effectiveness of the cache mechanism is based on the property of computer programs called *locality of reference*. Program analysis shows that most of their execution times is spent on routines where many instructions are executed repeatedly. These instructions may constitute simple loop, nested loop or a few procedures that repeatedly call each other. Locality of reference manifests itself in two ways: temporal and spatial. *Temporal* means that a recently executed instruction is likely to be executed again very soon. *Spatial* aspect means that instructions in close proximity to a recently executed instruction are likely to be executed soon. The concept here is to keep the active segments of the program in the fast cache so that total execution time can be reduced significantly as CPU accesses the cache for information and gets it faster. When the information required by the CPU is available in the cache, it is called a *hit*; otherwise it is referred to as a *miss*. When a miss happens, there are various algorithms discussed in later chapters that decide which block of information needs to be removed from the cache and a new a block of information is to be brought to the cache from the main memory. Importantly, the cache speeds up the processors. In superscalar CPUs, there are two caches—one for instructions and the other for data.

MMU (MEMORY MANAGEMENT UNIT)

The main memory cost is high and hence it has a limit on its size in a computer system. Large programs that need more main memory space than available, cannot run on this. Hence, the virtual-memory concept has been developed to solve this problem. The user need not know what the main memory size is and can assume very large memory size and write programs. The virtual-memory concept keeps the executable part of large program in the main memory and the remaining part in the secondary memory such as a magnetic disk. Whenever that part of the program that is to be executed is not in the main memory, the system will swap that part of the program from the disk and the not-required part of the program that is in the main memory goes to the disk. The information addresses generated for large programs are called *virtual addresses* and available main memory locations are *physical addresses*. **A Memory Management Unit (MMU)**, is responsible for handing accesses to *memory* requested by the *CPU*. Its functions include translation of *virtual addresses* to *physical addresses*.

BRANCH TARGET CACHE

Since a pre-fetch unit brings instructions to be executed in advance, if a branch instruction comes, the branch prediction foretells the outcome of the conditional branch instructions. The outcome of a branch instruction may result in flushing out of all pre-fetched instructions and getting a new set of instructions. Proper branch prediction will avoid frequent flushing of pre-fetched instructions and saves processor speed. BTC consists of a table with branch addresses, corresponding target addresses, and prediction information.

CONTROL UNIT

This is the central control unit controlling the functions of the computer system. There are two types of control units. One is conventional or hardwired control and the other one is microprogrammed control. The hardwired control unit is faster while the microprogrammed control unit gives greater flexibility in design that allows adding new instructions very easily. Current-day processors use both types of controls for executing the machine instructions.

INTEGER UNITS

This unit handles fixed-point binary arithmetic and logic operations. For binary arithmetic, 2's complement representation is used. All numbers are treated as integers.

FLOATING-POINT UNITS

This unit handles all floating-point arithmetic using IEEE standard single (32 bit), double (64 bit) and quad (128) precision floating-point binary numbers.

Both normalized and un-normalized floating-point arithmetic could be performed.

SPECIAL FUNCTIONAL UNITS (SFU)

Here, depending on the requirement of the special functional units such as fast Fourier transforms, Digital Signal Processing (DSP) can be incorporated.

Apart from these advances in the CPU, there are other advances such as pipelining, computer networking, and fast serial (gigabit) transmission in modern-day computers.

RISC AND CISC

There are two classes of computers called Complex Instruction Set Computer (CISC) and Reduced Instruction Set Computer (RISC). The differences between these two are given below:

	CISC	*RISC*
1.	Control unit is hardwired and microprogrammed	Control Unit is hardwired only
2.	Complex multiclock instructions	Simple single clock instructions
3.	Memory reference instructions	Register reference instructions except LOAD and STORE memory reference instructions
4.	Less registers in CPU	More in CPU
5.	Maximum memory addressing modes	Minimum memory addressing modes
6.	Small code sizes	Large code sizes
7.	Variable instruction length	Fixed instruction length

Most computers are designed with both RISC and CISC features. Hence they are niether RISC or CISC computers. Intel coined the word CRISC (Complex Reduced Instruction Set Computers) for such computers.

DEVELOPMENTS IN ELECTRONICS

Early computers were designed and fabricated using discrete active components such as vacuum tubes, transistors, small-scale integrated circuits, medium-scale integrated circuits and passive components such as resistors, inductors, capacitors, etc. As integrated technology developed as per Moore's law, which states that numbers of transistors on a single die doubles every two years (18 months), it resulted in VLSI circuits that has changed the scenario of computer fabrication. The sections below show the developments in integrated technology from SSI to VLSI.

Small-Scale Integration (SSI) — 1960–64

- Collection of one or more gates fabricated by only a single silicon chip.
- Small-scale integration contains an equivalent of 1 to 20 gates. They contain a handful of gates or flip-flops. Examples are logic gates such as NAND, NOR, Exclusive-OR, J_K flip-flops, D flip-flops, etc., (54/7400 series).
- SSI was introduced in the year 1960. In the 1960s, it was broadly based on bipolar transistors.
- Technology: 50–100 micrometer

Medium-Scale Integration (MSI)—1965–70

- Contains the equivalent of about 20–200 gates. An MSI typically contains function-building blocks such as a decoder, registers, or counters. This was introduced around 1965. Examples include ALU, Shift registers, counters, etc.
- Till recently, or even today, SSIs and MSIs are used in the 'glue' logic to interface together larger scale elements in complex systems.
- Technology: 10–50 micrometers.

Large-Scale Integration (LSI)—1970–76

- Contains the equivalent of 200 to 2000 gates or more.
- LSI part includes small memories, low-end microprocessors, programmable logic devices, etc.
- This was introduced in around 1970.
- In 1975, digital MOS ICs have prevailed. At present, even intrinsic speed advantage of bipolar transistors is being challenged by MOSFETs.
- Programmable logic devices were introduced in 1970.
- Field Programmable Gate Arrays (FPGAs) were introduced in 1990.

  Because of the advantages in device miniaturization, low power dissipation and high yield, it is expected that digital MOS ICs will dominate the IC market and capture a major market share of all semiconductor devices.
- Bipolar transistor technology has shown more capacity for improvement than expected and is still an important factor.
- Another trend is the gradual introduction of GaAs (Gallium Arsenide) digital integrated technology for selected applications.
- Technology: 5–10 micrometers.

Very Large-Scale Integration (VLSI)—1976

- The dividing line between LSI and VLSI (Very Large-Scale Integrated Circuits) is not clear and tends to be stated in terms of transistor count rather than gate count.
- An IC with over 1, 000, 000 transistors is definitely VLSI and includes most microprocessors and memories of today.
- This VLSI started in the year 1975.
- Technology: < 2 micrometers.

Ultra Large Scale Integration (ULSI)

- When the transistor count increases to a million transistors, the technology is referred to as ULSI—Ultra Large-Scale Integrated Circuits.
- Technology: < 1 micrometer.

  Hence, the integrated technology started with 50–100 micrometers in 1960 and now it is 28 nanometers with billions of transistors on a single chip. This tremendous technological development has given rise to superscalar processors and muticore processors on a single chip.

MICROPROCESSOR EVOLUTION

Because of the dramatic developments in semiconductor technology, computers that were fabricated with discrete components have started using silicon chips. These VLSI circuits gave way for the development of fixed instruction set microprocessors, bit-slice processors and microcontrollers on a single silicon die. The microprocessor is a CPU on a single chip. Bit-slice processors are Register-ALU (RALU) on a chip and micro-instruction sequencer on another chip. These two sets of chips could be used to build a computer of required word length. To build a computer using a microprocessor, one should select memories, a memory controller, a peripheral interface and other components and interconnect them using a system bus. A microcontroller is a computer with the CPU, limited program and data memory and I/O interfaces, timers, etc., on a single chip.

CPUs started with 4-bit processor chips and have gradually grown into 8-bit, 16-bit, 32-bit and 64-bit Processors (CPU) with all required support components in LSI/VLSI technology such as memories and peripheral interfaces to build high-end computer systems. Microprocessors are manufactured by many companies initially but Intel dominated this field and continues to manufacture superscalar and multicore microprocessors. Other companies have gradually entered into this field. It is interesting to see the evolution of Intel Microprocessors as shown below.

1971

Intel introduced its 4-bit bus, 4004 chip—the first microprocessor. Speed was 60, 000 operations per second. It used 2300 transistors, based on 10-micron technology and addressed 640 bytes. The die for the chip measures 3 × 4 mm. It operated at 750 kHz.

1972

Intel introduced its 8008 chip, the first 8-bit microprocessor. It accessed 16 KB of memory, used 3500 transistors, based on 10-micron technology. Its speed was 60,000 instructions per second.

1974

Intel released its 2 MHz 8080 chip, an 8-bit microprocessor. It could access 64 KB of memory used 6000 transistors, based on 6-micron technology with a speed of 0.64 MIPS. This processor used two power supplies. Intel introduced the single power supply 8-bit processor 8085 in 1976 which is still widely used.

1978

Intel introduced the 4.77 MHz 8086 microprocessor. It used 16-bit registers, a 16-bit data bus, and 29,000 transistors, using 3-micron technology and could access 1 MB of memory. Its speed was 0.33 MIPS. Later speeds included 8 MHz (0.66 MIPS) and 10 MHz (0.75 MIPS). This 8086 has a co-math processor 8087 introduced in 1980 to do floating-point arithmetic. Other mathematical operation advances in computer architecture started from this processor.

1979

Intel introduced the 4.77 MHz 8088 microprocessor which was created as a stepping stone to the 8086, as it operates on 16 bits internally, but supports an 8-bit data bus, to use existing 8-bit device-controlling chips. It contained 29, 000 transistors, using 3-micron technology, and could address 1 MB of memory. Its speed was 0.33 MIPS. A later version operates at 8 MHz, for a speed of 0.75 MIPS.

1985

Intel introduced the 6 MHz 80286 microprocessor and used a 16-bit data bus, 134, 000 transistors using 1.5 micron technology and offers protected mode operation. It could access 16 MB of memory or 1 GB of virtual memory. Its speed was 0.9 MIPS. Later versions operate at 10 MHz 1.5 MIPS, and 12 MHz 2.66 MIPS.

1985

Intel introduced the 16 MHz 80386DX microprocessor. It used 32-bit registers and a 32-bit data bus, and incorporated 275, 000 transistors (1.5 microns). It could access 4 gigabytes of physical memory.

1989

Intel announced the 25 MHz microprocessor at Spring Comdex in Chicago, Illinois. It integrated the 386, 387 math coprocessor, and added an 8 KB primary cache. It used 1.2 million transistors, employing 1-micron technology at a speed of 20 MIPS.

1991

Intel introduced the 50 MHz 486 microprocessor. Speed was 41 MIPS. This new 486 employs 0.8-micron technology.

1992

Intel introduced the 486SL processor, designed for notebook computers. Speeds include 20 MHz (15.4 MIPS), 25 MHz (19 MIPS) and 33 MHz (25 MIPS). The processors can address 64 MB of physical memory. They used 1.4 million transistors, employing 0.8-micron technology.

1993

Intel introduced the Pentium processor and used 32-bit registers, with a 64-bit data bus, giving it an address space of 4 GB. It incorporated 3.1 million transistors, using 0.8-micron BiCMOS technology. Speeds are 60 MHz (100 MIPS) and 66 MHz (112 MIPS).

1994

Intel introduced the 75 MHz Pentium processor. Speed was 126. 5 MIPS. It uses 3.2 million transistors, employing 0.6-micron BiCMOS technology.

1995

Intel released the Pentium Pro which contained 5.5 million transistors.

1995

Intel announced the immediate availability of the 133 MHz Pentium Processor. It used 3.2 million transistors, employing 0.35 micron BiCMOS technology. Speed was 218.9 MIPS.

1996

In the 133 MHz Pentium processor for notebook computers, the processor used 0.35 micron technology, and operated on 3.3 volts of power externally, while its internal core only required 2.9 volts.

1996

Intel released the 150 MHz mobile Pentium processor, designed for use in portable computers. The processor used 0.35-micron technology, and operated on 3.3 volts of power externally, while its internal core only requires 3.1 volts.

1997

Intel released the 7.7 million transistor Pentium II processor. This processor uses 0.25 micron technology.

1999

Intel released the 450 MHz Pentium III Processor. This processor uses 0.18 micron technology and operates from 450 to 1.13 GHz frequency.

2000

Intel released Pentium IV operating at 2 GHz and there is a Pentium 4 family of processors.

2005

In 2005, Intel released its first desktop *dual-core processor* (two independent processor cores in a single package) called Pentium D. The number of transistors in the Pentium D processor is 291 million and 3.2 GHz initial speed. Pentium extreme edition is also the other dual-core processor, referred to as the high-end version of Pentium D CPU. Unlike the Pentium D, it supports *hyper-threading technology* and overclocking.

Present Status

All of the currently available and upcoming Intel performance desktop processors use Pentium 4 technology. The 'Tejas' project which was intended to produce the 'Pentium 5' was discontinued and abandoned earlier in 2004, and just lately Intel has also abandoned plans to produce a 4 GHz Pentium 4 processor.

Figure 1.11 show how Moore's law was applicable to Intel processors.

OTHER WIDELY USED MICROPROCESSORS

Motorola

6800—	Widely used 8-bit processors in 1974
68000—	Widely used internally 32-bit but with 16-bit external bus in 1979.
68010—	An enhanced version of the 68000 released in 1982
68020—	32-bit processor in 1984
68030—	An enhanced version of 68020 released in 1987
68040—	A successor to the 68030 released in 1989

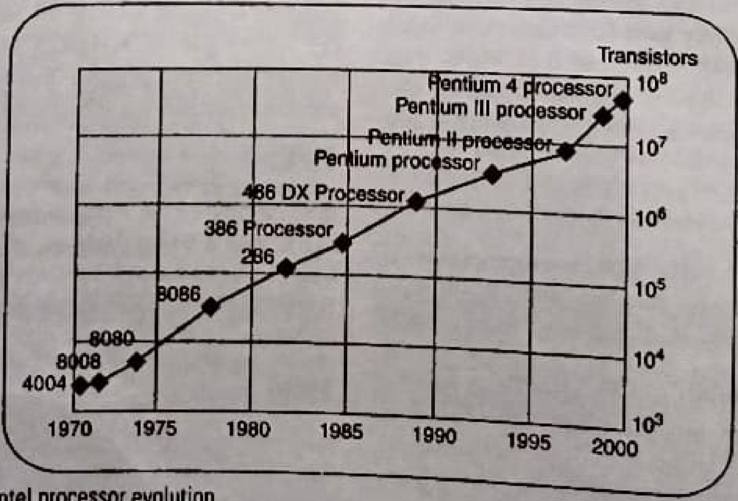

Fig. 1.11 Moore's Law for Intel processor evolution

Zilog

Z-80—Very widely used 8-bit processor released in 1976
Z-8000—16-bit processor released in 1979

Fairchild

F 8—8-bit processor released in 1975

In this chapter, evolutions of generations of computers, developments in electronics and microprocessors and advances in computer architecture are briefly discussed to give the reader a better understanding of the field of microprocessors. The next chapter discusses in detail about an 8-bit Microprocessor 8085 and a 16-bit processor 8086 architectures.

Checklist of Important Terms and Concepts in This Chapter

- *Generations of computers*
- *Von Neumann and Harvard architectures*
- *Hardware, software, operating system, computer languages*
- *Batch processing, multiprogramming, timesharing*
- *Instruction formats, addressing modes*
- *Fixed-point, floating-point data formats*
- *Interrupts ISR, interrupt response time*
- *ASCII, EBCDIC*
- *Bus*
- *DMA*
- *Cache*
- *MMU*
- *RISC, CISC*
- *Moore's law*
- *SSI, MSI, LSI, VLSI*
- *Microprocessor evolution*

Review Questions and Problems

*1. Why can a binary state device be built reliably using an active component and not a device with ten distinguishable states?

*2. What are the advantages of 2's complement representation of binary signed numbers over other representations?

*3. In an 8-bit register, the following signed binary numbers in 2's complement representation are stored. Give their decimal equivalents:
   i) 11010110   ii) 01101111
   iii) 10000000   iv) 01000001

**4. In von Neuman architecture, how does the CPU recognize the information it received from primary memory as an instruction or data?

**5. From input unit to CPU or memory and CPU or memory to output unit, in what format is information is transferred?

*6. Represent the decimal number $A = -187$ in fixed point
   (a) Signed magnitude form
   (b) Signed 1's complement form
   (c) Signed 2's complement form

**7. Represent the decimal number $A = 187.52$ in single precision binary biased exponent floating-point representation.

*8. Represent $A = 187$ in ASCII zoned format using 8-4-2-1 BCD.

*9. Name some input only, output only and input and output devices in a computer system.

**10. When an interrupt comes, why does not the CPU immediately start the interrupt service routine, instead of completing the execution of ongoing instruction and start servicing the interrupt service routine?

**11. How do you compute the interrupt response time of a computer?

***12. In programmed interrupt and DMA I/O modes, which device with maximum speed of data transfer can be handled and what could be the speed of that device?

***13. What is virtual memory and what function is performed by the memory management unit?

**14. How many guard bits would be necessary in a normalized floating point unit to maintain maximum precision of floating-point arithmetic and why?

**15. What do you think are the problems that designers and manufacturers may be facing in placing billions of transistors on a single die?