LECTURE NOTES

ON

Digital Logic Design and Computer Organization

Department of Information Technology

UNIT-1 PART-1

Basic Structure of Computers:

1.1.1Computer Types

1.1.2 Functional units

1.1.3 Basic operational concepts

1.1.4 Bus structures

1.1.5 Software

1.1.6 Performance

1.1.7 multiprocessors and multi computers

1.1.8 Computer Generations.

Data Representation: Binary Numbers, Fixed Point Representation .Floating Point Representation. Number base conversions, Octal and Hexadecimal Numbers, components, Signed binary numbers, Binary codes.

1.1.1 Computer types

A computer can be defined as a fast electronic calculating machine that accepts the (data) digitized input information process it as per the list of internally stored instructions and produces the resulting information.

List of instructions are called programs & internal storage is called computer memory.

The different types of computers are

1. **Personal computers: -** This is the most common type found in homes, schools, Business offices etc., It is the most common type of desk top computers with processing and storage units along with various input and output devices.
2. **Note book computers: -** These are compact and portable versions of PC
3. **Work stations: -** These have high resolution input/output (I/O) graphics capability, but with same dimensions as that of desktop computer. These are used in engineering applications of interactive design work.
4. **Enterprise systems: -** These are used for business data processing in medium to large corporations that

require much more computing power and storage capacity than work stations. Internet associated with

servers have become a dominant worldwide source of all types of information.

5. **Super computers: -** These are used for large scale numerical calculations required in the applications like weather forecasting etc.,

1.1.2. Functional unit

A computer consists of five functionally independent main parts input, memory, arithmetic logic unit (ALU), output and control unit.

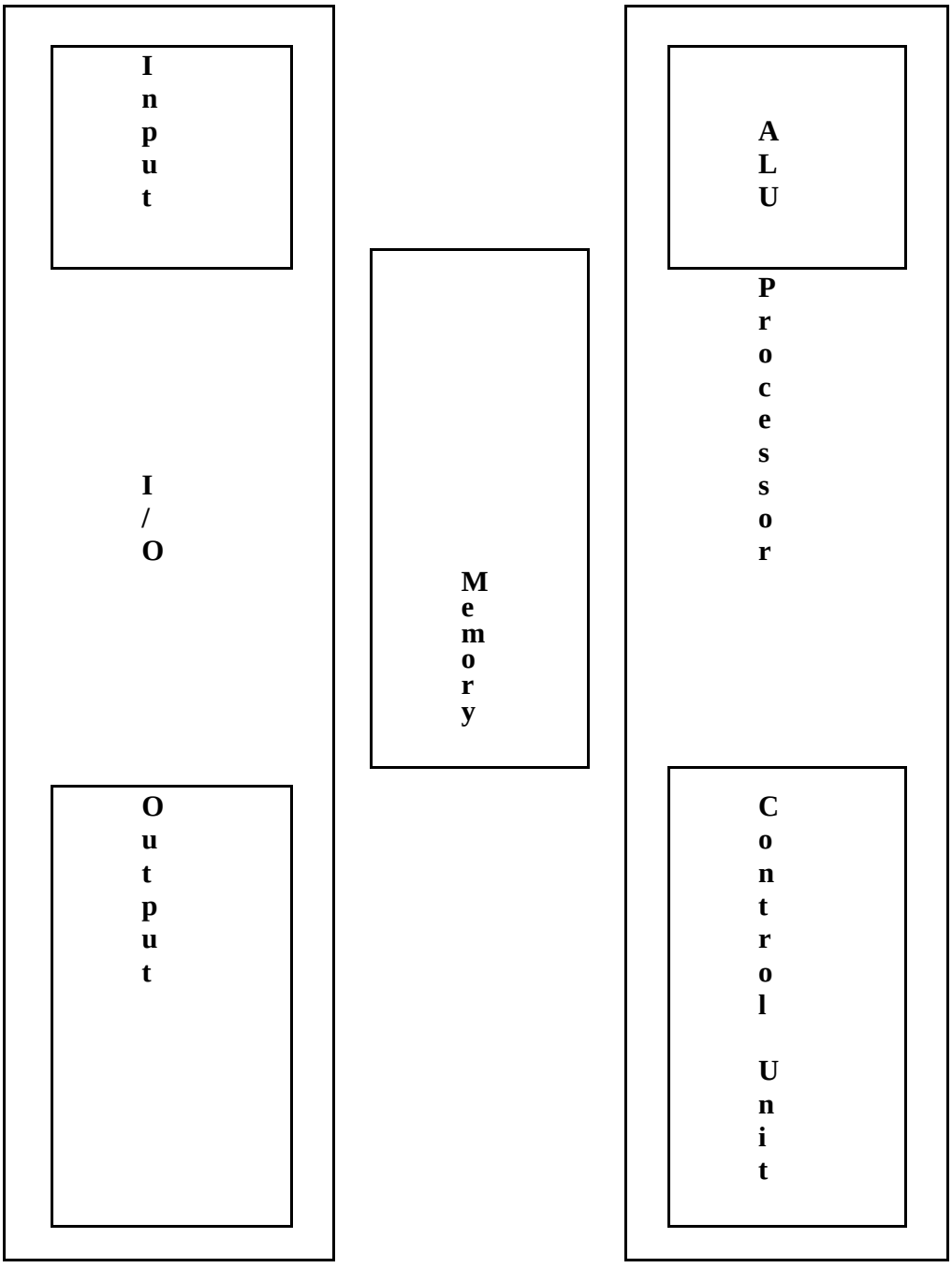


Fig a : Functional units of computer

Input device accepts the coded information as source program i.e. high level language. This is either stored in the memory or immediately used by the processor to perform the desired operations. The program stored in the memory determines the processing steps. Basically the computer converts one source program to an object program. i.e. into machine language.

Finally the results are sent to the outside world through output device. All of these actions are coordinated by the control unit.

Input unit: -

The source program/high level language program/coded information/simply data is fed to a computer through input devices keyboard is a most common type. Whenever a key is pressed, one corresponding word or number is translated into its equivalent binary code over a cable & fed either to memory or processor.

Joysticks, trackballs, mouse, scanners etc are other input devices.

Memory unit: -

Its function into store programs and data. It is basically to two types

6. **Primary memory**
7. **Secondary memory**

Primary memory: - Is the one exclusively associated with the processor and operates at the electronics speeds

programs must be stored in this memory while they are being executed. The memory contains a large number of semiconductors storage cells. Each capable of storing one bit of information. These are processed in a group of fixed site called word.

To provide easy access to a word in memory, a distinct address is associated with each word location. **Addresses are** numbers that identify memory location.

Number of bits in each word is called word length of the computer. Programs must reside in the memory during execution. Instructions and data can be written into the memory or read out under the control of processor.

Memory in which any location can be reached in a short and fixed amount of time after specifying its address is called random-access memory (RAM).

The time required to access one word in called memory access time. Memory which is only readable by the user and contents of which can't be altered is called read only memory (ROM) it contains operating system.

Caches are the small fast RAM units, which are coupled with the processor and are aften contained on the same IC chip to achieve high performance. Although primary storage is essential it tends to be expensive.

2 Secondary memory: - Is used where large amounts of data & programs have to be stored, particularly information that is accessed infrequently.

Examples: - Magnetic disks & tapes, optical disks (ie CD-ROM's), floppies etc.,

Arithmetic logic unit (ALU):-

Most of the computer operators are executed in ALU of the processor like addition, subtraction, division, multiplication, etc. the operands are brought into the ALU from memory and stored in high speed storage elements called register. Then according to the instructions the operation is performed in the required sequence.

The control and the ALU are may times faster than other devices connected to a computer system. This enables a single processor to control a number of external devices such as key boards, displays, magnetic and optical disks, sensors and other mechanical controllers.

Output unit:-

These actually are the counterparts of input unit. Its basic function is to send the processed results to the outside world.

Examples:- Printer, speakers, monitor etc.

Control unit:-

It effectively is the nerve center that sends signals to other units and senses their states. The actual timing signals that govern the transfer of data between input unit,

processor, memory and output unit are generated by the control unit.

1.1.3 Basic operational concepts

To perform a given task an appropriate program consisting of a list of instructions is stored in the memory.

Individual instructions are brought from the memory into the processor, which executes the specified operations. Data to be stored are also stored in the memory.

Examples: - Add LOCA, R_0

This instruction adds the operand at memory location LOCA, to operand in register R_0 & places the sum into register. This instruction requires the performance of several steps,

1.First the instruction is fetched from the memory into the processor. 2.The operand at LOCA is fetched and added to the contents of R_0

3.Finally the resulting sum is stored in the register R_0

The preceding add instruction combines a memory access operation with an ALU Operations. In some other type of computers, these two types of operations are performed by separate instructions for performance reasons.

Load LOCA, R1 Add R1, R0

Transfers between the memory and the processor are started by sending the address of the memory location to be accessed to the memory unit and issuing the appropriate control signals. The data are then transferred to or from the memory.

The fig shows how memory & the processor can be connected. In addition to the ALU & the control circuitry, the processor contains a number of registers used for several different purposes.

The instruction register (IR):- Holds the instructions that is currently being executed. Its output is available for the control circuits which generates the timing signals that control the various processing elements in one execution of instruction.

The program counter PC:-

This is another specialized register that keeps track of execution of a program. It contains the memory address of the next instruction to be fetched and executed.

The other two registers which facilitate communication with memory are: -

1.MAR – (Memory Address Register):- It holds the address of the location to be accessed.

2.MDR – (Memory Data Register):- It contains the data to be written into or read out of the address location.

Operating steps are

1.Programs reside in the memory & usually get these through the I/P unit.

2. Execution of the program starts when the PC is set to point at the first instruction of the program. 3.Contents of PC are transferred to MAR and a Read Control Signal is sent to the memory.

4. After the time required to access the memory elapses, the address word is read out of the memory and loaded into the MDR.

2. Now contents of MDR are transferred to the IR & now the instruction is ready to be decoded and executed.
3. If the instruction involves an operation by the ALU, it is necessary to obtain the required operands.
4. An operand in the memory is fetched by sending its address to MAR & Initiating a read cycle.
5. When the operand has been read from the memory to the MDR, it is transferred from MDR to the ALU.
6. After one or two such repeated cycles, the ALU can perform the desired operation.
7. If the result of this operation is to be stored in the memory, the result is sent to MDR.
8. Address of location where the result is stored is sent to MAR & a write cycle is initiated.
9. The contents of PC are incremented so that PC points to the next instruction that is to be executed.

Normal execution of a program may be preempted (temporarily interrupted) if some devices require urgent servicing, to do this one device raises an Interrupt signal.

An interrupt is a request signal from an I/O device for service by the processor. The processor provides the requested service by executing an appropriate interrupt service routine.

The Diversion may change the internal stage of the processor its state must be saved in the memory location before interruption. When the interrupt-routine service is completed the state of the processor is restored so that the interrupted program may continue.

1.1.4 Bus structure

The simplest and most common way of interconnecting various parts of the

computer. To achieve a reasonable speed of operation, a computer must be organized so that all its units can handle one full word of data at a given time.A group of lines that serve as a connecting port for several devices is called a bus.

In addition to the lines that carry the data, the bus must have lines for address and control purpose. Simplest way to interconnect is to use the single bus as shown

Since the bus can be used for only actively use the bus at any given time. Bus requests for use of one bus, one transfer at a time, only two units can control lines are used to arbitrate multiple

Single bus structure is Low cost
Very flexible for attaching peripheral devices

Multiple bus structure certainly increases, the performance but also increases the cost significantly.

All the interconnected devices are not of same speed & time, leads to a bit of a problem. This is solved by using cache registers (ie buffer registers). These buffers are electronic registers of small capacity when compared to the main memory but of comparable speed.

The instructions from the processor at once are loaded into these buffers and then the complete transfer of data at a fast rate will take place.

1.1.5 Performance

The most important measure of the performance of a computer is how quickly it can execute programs. The speed with which a computer executes program is affected by the design of its hardware. For best performance, it is necessary to design the compiles, the machine instruction set, and the hardware in a coordinated way.

The total time required to execute the program is elapsed time is a measure of the performance of the entire computer system. It is affected by the speed of the processor, the disk and the printer. The time needed to execute a instruction is called the processor time.

Just as the elapsed time for the execution of a program depends on all units in a computer system, the processor time depends on the hardware involved in the execution of individual machine instructions. This hardware comprises the processor and the memory which are usually connected by the bus as shown in the fig c.

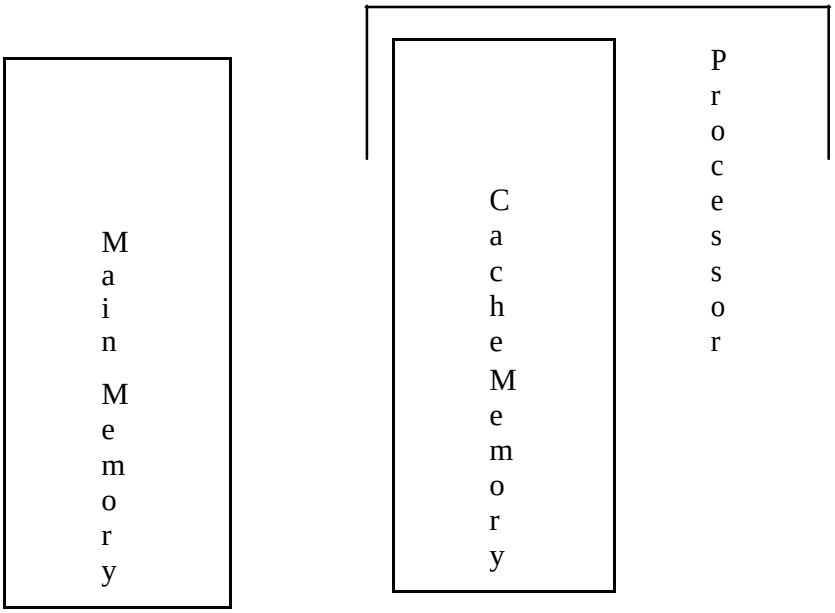


Fig d: The processor cache

Let us examine the flow of program instructions and data between the memory and the processor. At the start of execution, all program instructions and the required data are stored in the main memory. As the execution proceeds, instructions are fetched one by one over the bus into the processor, and a copy is placed in the cache later if the same instruction or data item is needed a second time, it is read directly from the cache.

The processor and relatively small cache memory can be fabricated on a single IC chip. The internal speed of performing the basic steps of instruction processing on chip is very high and is considerably faster than the speed at which the instruction and data can be fetched from the main memory. A program will be executed faster if the

movement of instructions and data between the main memory and the processor is minimized, which is achieved by using the cache.

For example:- Suppose a number of instructions are executed repeatedly over a short period of time as happens in a program loop. If these instructions are available in the cache, they can be fetched quickly during the period of repeated use. The same applies to the data that are used repeatedly.

Processor clock: -

Processor circuits are controlled by a timing signal called clock. The clock designer the regular time intervals called clock cycles. To execute a machine instruction the processor divides the action to be performed into a sequence of basic steps that each step can be completed in one clock cycle. The length P of one clock cycle is an important parameter that affects the processor performance.

Processor used in today's personal computer and work station have a clock rates that range from a few hundred million to over a billion cycles per second.

1.1.6 Basic performance equation

We now focus our attention on the processor time component of the total elapsed time. Let 'T' be the processor time required to execute a program that has been prepared in some high-level language. The compiler generates a machine language object program that corresponds to the source program. Assume that complete execution of the program requires the execution of N machine cycle language instructions. The number N is the actual number of instruction execution and is not necessarily equal to the number of machine cycle instructions in the object program. Some instruction may be executed more than once, which in the case for instructions inside a program loop others may not be executed all, depending on the input data used

Suppose that the average number of basic steps needed to execute one machine cycle instruction is S, where each basic step is completed in one clock cycle. If clock rate is 'R' cycles per second, the program execution time is given by

$$T = \frac{N \times S}{R}$$

this is often referred to as the basic performance equation.

We must emphasize that N, S & R are not independent parameters changing one may affect another. Introducing a new feature in the design of a processor will lead to improved performance only if the overall result is to reduce the value of T.

Pipelining and super scalar operation: -

We assume that instructions are executed one after the other. Hence the value of S is the total number of basic steps, or clock cycles, required to execute one instruction. A substantial improvement in performance can be achieved by overlapping the execution of successive instructions using a technique called pipelining.

Consider Add R_1 R_2 R_3

This adds the contents of R_1 & R_2 and places the sum into R_3.

The contents of R_1 & R_2 are first transferred to the inputs of ALU. After the addition operation is

performed, the sum is transferred to R_3. The processor can read the next instruction from the memory, while the addition operation is being performed. Then of that instruction also uses, the ALU, its operand can be transferred to the ALU inputs at the same time that the add instructions is being transferred to R_3.

In the ideal case if all instructions are overlapped to the maximum degree possible the execution proceeds at

the rate of one instruction completed in each clock cycle. Individual instructions still require several clock cycles to complete. But for the purpose of computing T, effective value of S is 1.

A higher degree of concurrency can be achieved if multiple instructions pipelines are implemented in the processor. This means that multiple functional units are used creating parallel paths through which different instructions can be executed in parallel with such an arrangement, it becomes possible to start the execution of several instructions in every clock cycle. This mode of operation is called superscalar execution. If it can be sustained for a long time during program execution the effective value of S can be reduced to less than one. But the parallel execution must preserve logical

correctness of programs, that is the results produced must be same as those produced by the serial execution of program instructions. Now a days may processor are designed in this manner.

1.1.7 Clock rate

These are two possibilities for increasing the clock rate 'R'.

1. Improving the IC technology makes logical circuit faster, which reduces the time of execution of basic steps. This allows the clock period P, to be reduced and the clock rate R to be increased.

2. Reducing the amount of processing done in one basic step also makes it possible to reduce the clock period P. however if the actions that have to be performed by an instructions remain the same, the number of basic steps needed may increase.

Increase in the value 'R' that are entirely caused by improvements in IC technology affects all aspects of the processor's operation equally with the exception of the time it takes to access the main memory. In the presence of cache the percentage of accesses to the main memory is small. Hence much of the performance gain excepted from the use of faster technology can be realized.

Instruction set CISC & RISC:-

Simple instructions require a small number of basic steps to execute. Complex instructions involve a large number of steps. For a processor that has only simple instruction a large number of instructions may be needed to perform a given programming task. This could lead to a large value of 'N' and a small value of 'S' on the other hand if individual instructions perform more complex operations, a fewer instructions will be needed, leading to a lower value of N and a larger value of S. It is not obvious if one choice is better than the other.

But complex instructions combined with pipelining (effective value of S ¿ 1) would achieve one best performance. However, it is much easier to implement efficient pipelining in processors with simple instruction sets.

1.1.8 Performance measurements

It is very important to be able to access the performance of a computer, comp designers use performance estimates to evaluate the effectiveness of new features.

The previous argument suggests that the performance of a computer is given by the

execution time T, for the program of interest.

Inspite of the performance equation being so simple, the evaluation of 'T' is highly complex. Moreover the parameters like the clock speed and various architectural features are

not reliable indicators of the expected performance.

Hence measurement of computer performance using bench mark programs is done to make comparisons possible, standardized programs must be used.

The performance measure is the time taken by the computer to execute a given bench mark. Initially some attempts were made to create artificial programs that could be used as bench mark programs. But synthetic programs do not properly predict the performance obtained when real application programs are run.

A non profit organization called SPEC- system performance evaluation corporation selects and publishes bench marks.

The program selected range from game playing, compiler, and data base applications to numerically intensive programs in astrophysics and quantum chemistry. In each case, the program is compiled under test, and the running time on a real computer is measured. The same program is also compiled and run on one computer selected as reference.

The 'SPEC' rating is computed as follows.

Running time on the reference computer

SPEC rating =

Running time on the computer under test If the SPEC rating =

50

Means that the computer under test is 50 times as fast as the ultra sparc 10. This is repeated for all the programs in the SPEC suit, and the geometric mean of the result is computed.

Let SPECi be the rating for program 'i' in the suite. The overall SPEC rating for the computer is given by

SPEC rating = $\left(\pi_{i=1}^{n} SPECi\right)$

Where 'n' = number of programs in suite.

Since actual execution time is measured the SPEC rating is a measure of the combined effect of all factors affecting performance, including the compiler, the OS, the processor, the memory of comp being tested.

1.1.9. Multiprocessor & microprocessors:-

Large computers that contain a number of processor units are

called multiprocessor system.

These systems either execute a number of different application tasks in parallel or execute subtasks of a single large task in

parallel.

All processors usually have access to all memory locations in such system & hence they are

called shared memory multiprocessor systems.

The high performance of these systems comes with much

increased complexity and cost.

In contrast to multiprocessor systems, it is also possible to use an interconnected group of complete computers to achieve high total computational power. These computers normally have access to their own memory units when the tasks they are executing need to communicate data they do so by exchanging messages over a communication network. This properly distinguishes them from shared memory multiprocessors, leading to name message-passing multi computer.

BIG-ENDIAN AND LITTLE-ENDIAN ASIGNMENTS:-

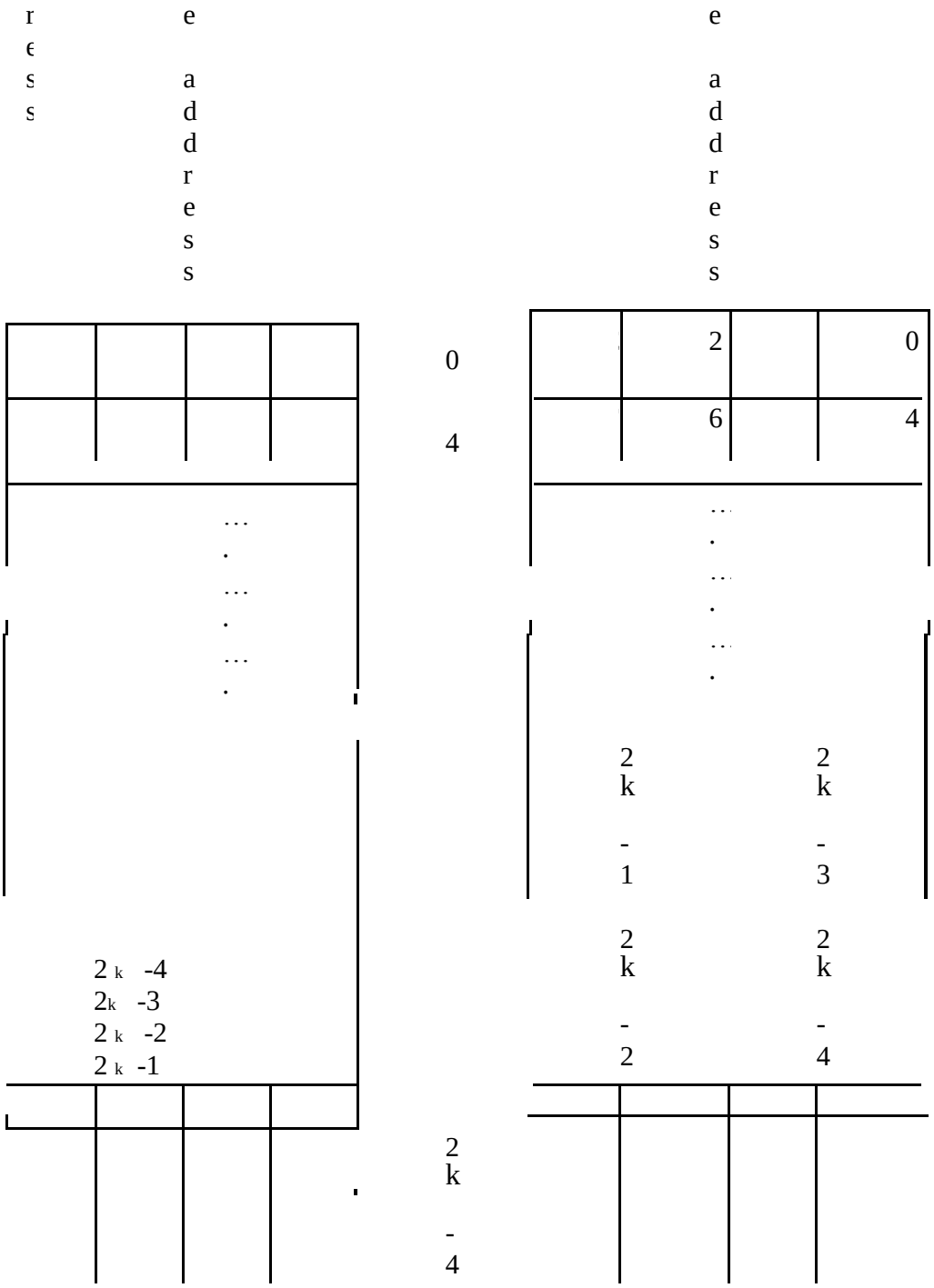
There are two ways that byte addresses can be assigned across words, as shown in fig b. The name big-endian is used when lower byte addresses are used for the more significant bytes (the leftmost bytes) of the word. The name little-endian is used for the opposite ordering, where the lower byte addresses are used for the less significant bytes (the rightmost bytes) of the word.

In addition to specifying the address ordering of bytes within a word, it is also necessary to specify the labeling of bits within a byte or a word. The same ordering is

also used for labeling bits within

a byte, that is, $b_7, b_6, \ldots, b_0$, from left to right. Word

A	B	B
c	y	y
c	t	t



(a) Big-**endian** **a**ssignment

(b) Little-endian assignment

1.1.10. BUSES:

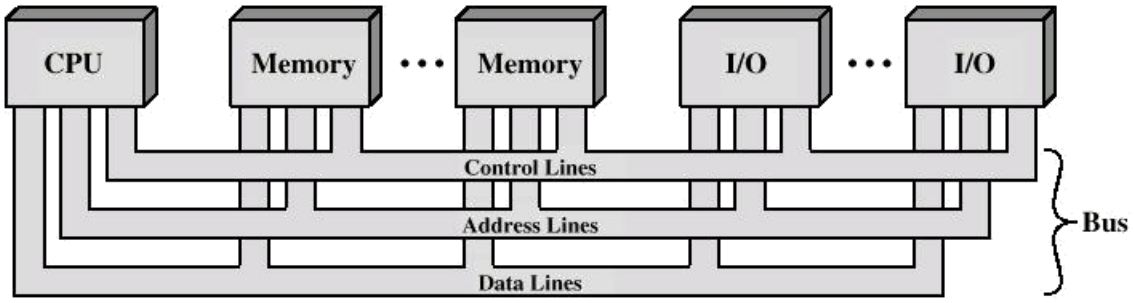
- There are a number of possible interconnection systems
- Single and multiple BUS structures are most common
- e.g. Control/Address/Data bus (PC)
- e.g. Unibus (DEC-PDP)

WHAT IS A BUS

- A communication pathway connecting two or more devices
- Usually broadcast (all components see signal)
- Often grouped
- A number of channels in one bus
- e.g. 32 bit data bus is 32 separate single bit channels
- Power lines may not be shown
-

BUS INTER CONNECTION

SCHEME



DATA BUS:

- Carries data

— Remember that there is no difference between “data” and “instruction” at this level

- Width is a key determinant of performance

— 8, 16, 32, 64 bit

ADDRESS BUS:

- Identify the source or destination of data

- e.g. CPU needs to read an instruction (data) from a given location in memory

- Bus width determines maximum memory capacity of system

— e.g. 8080 has 16 bit address bus giving 64k address space

CONTROL BUS:

- Control and timing information

— Memory read/write signal

— Interrupt request Clock signals

What do buses look like?

— Parallel lines on circuit boards

— Ribbon cables

— Strip connectors on mother boards

– e.g. PCI Sets of wires

SINGLE BUS PROBLEMS:

- Lots of devices on one bus leads to:

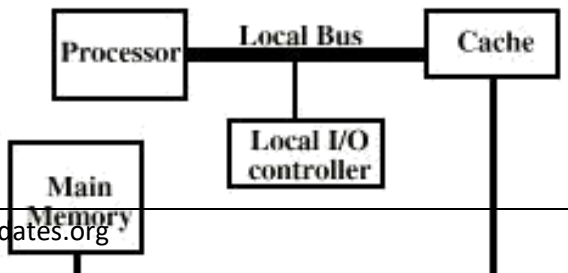
— Propagation delays

– Long data paths mean that co-ordination of bus use can adversely affect performance

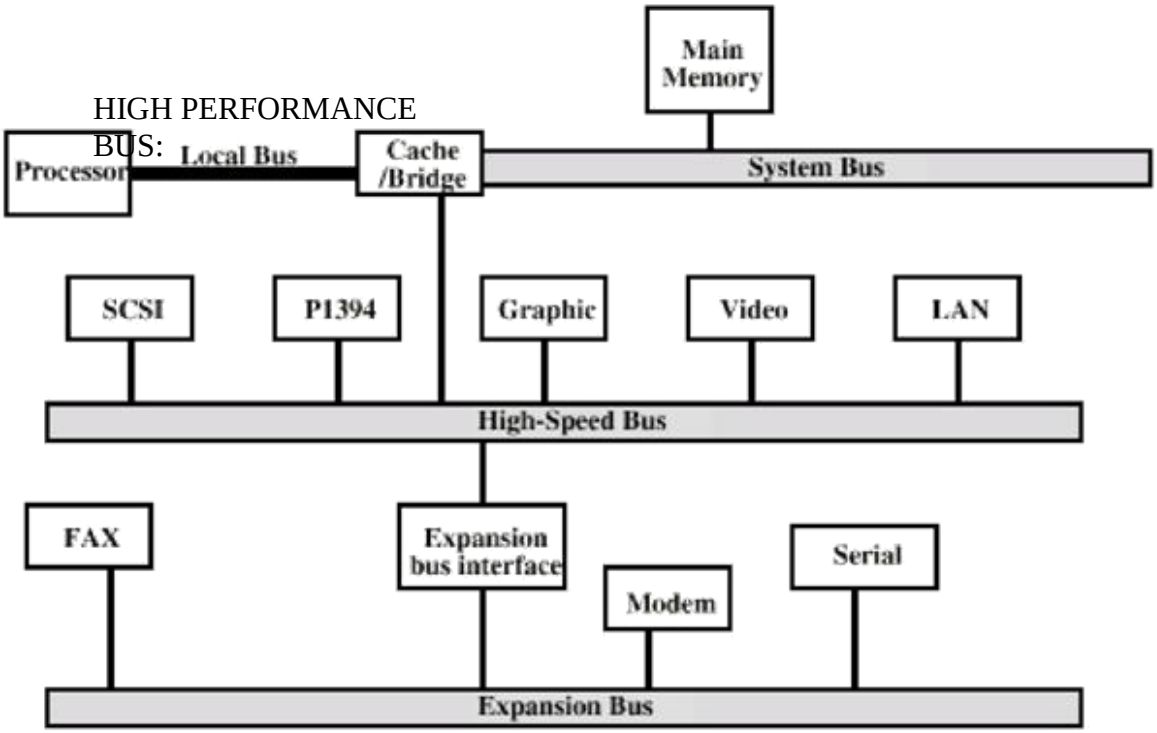
– If aggregate data transfer approaches bus capacity

- Most systems use multiple buses to overcome these problems

TRADITIONAL ISA WITH CACHE:



HIGH PERFORMANCE BUS:



BUS TYPES:

- Dedicated

  — Separate data & address lines

- Multiplexed

  — Shared lines

  — Address valid or data valid control line

  — Advantage - fewer lines

  — Disadvantages

  – More complex control

  – Ultimate performance

BUS ARBITRATION:

- More than one module controlling the bus

— e.g. CPU and DMA controller

- Only one module may control bus at one time
    - Arbitration may be centralised or distributed

CENTRALIZED ARBITRTAION:

- Single hardware device controlling bus access

— Bus Controller

— Arbiter

• May be part of CPU or separate

DISTRIBUTED ARBITRATION:

- Each module may claim the bus
  - Control logic on all modules PCI BUS:
- Peripheral Component Interconnection (PCI)
- Intel released to public domain
- 32 or 64 bit
- 50 lines

PCI BUS LINES(REQUIRED)

- Systems lines
  - — Including clock and reset
- Address & Data
  - — 32 time mux lines for address/data
  - — Interrupt & validate lines
- Interface Control
- Arbitration
  - — Not shared
  - — Direct connection to PCI bus arbiter
- Error lines

PCI BUS LINES (OPTIONAL)

- Interrupt lines
  - — Not shared
- Cache support
- 64-bit Bus Extension
  - — Additional 32 lines
  - — Time multiplexed
  - — 2 lines to enable devices to agree to use 64-bit transfer
- JTAG/Boundary Scan

For testing procedures

PCI COMMANDS:

- Transaction between initiator (master) and target
- Master claims bus
- Determine type of transaction
  — e.g. I/O read/write
- Address phase
- One or more data phases

MULTIPROCESSOR AND MULTICOMPUTERS

Two categories of parallel computers are discussed below namely shared common memory or unshared distributed

memory.

Shared memory multiprocessors

Shared memory parallel computers vary widely, but generally have in common the ability for all processors to access all

memory as global address space.

- Multiple processors can operate independently but share the same memory resources.
- Changes in a memory location effected by one processor are visible to all other processors.
- Shared memory machines can be divided into two main classes based upon memory access times: UMA , NUMA and COMA. Uniform Memory Access (UMA):
- Most commonly represented today by Symmetric Multiprocessor (SMP) machines
- Identical processors
- Equal access and access times to memory
- Sometimes called CC-UMA - Cache Coherent UMA. Cache coherent means if one processor updates a location in shared memory, all the other processors know about the update. Cache coherency is accomplished at the hardware level.

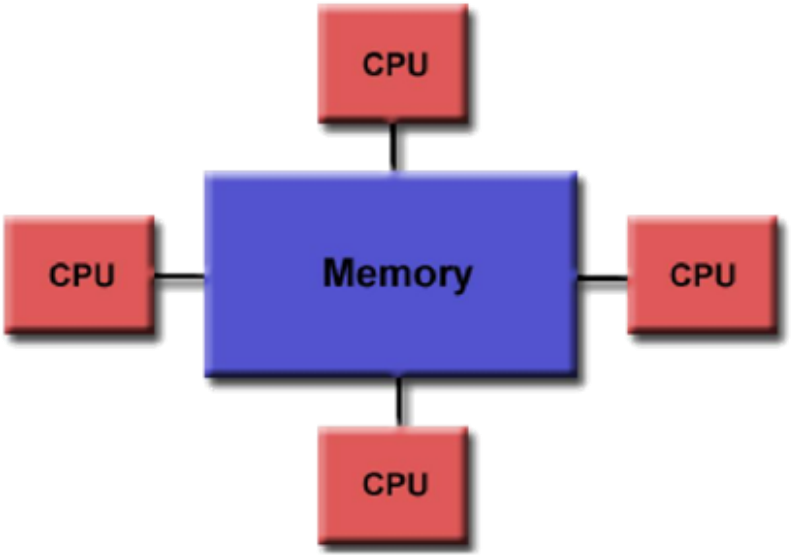


Figure 1.9 Shared Memory (UMA)

Non-Uniform Memory Access (NUMA):

- Often made by physically linking two or more SMPs
- One SMP can directly access memory of another SMP
- Not all processors have equal access time to all memories
- Memory access across link is slower

If cache coherency is maintained, then may also be called CC-NUMA - Cache Coherent NUMA

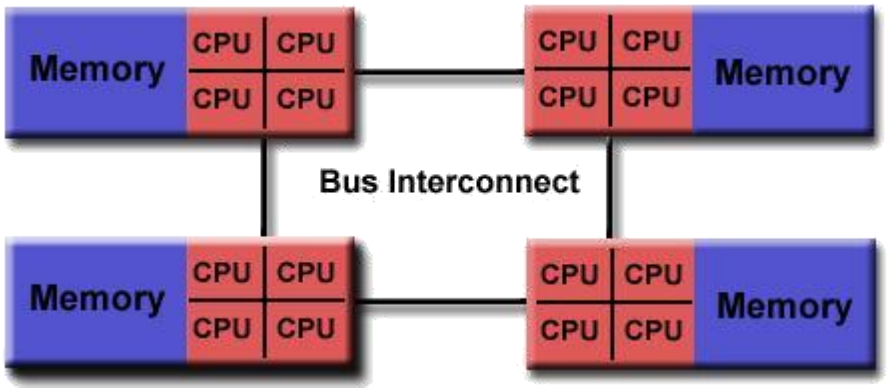


figure 1.10 **Shared Memory (NUMA)**

The COMA model : The COMA model is a special case of NUMA machine in which the distributed main memories are converted to caches. All caches form a global address space and there is no memory hierarchy at each processor node.

Advantages:

- Global address space provides a user-friendly programming perspective to memory
- Data sharing between tasks is both fast and uniform due to the proximity of memory to CPUs

Disadvantages:

- Primary disadvantage is the lack of scalability between memory and CPUs. Adding more CPUs can geometrically increases traffic on the shared memory CPU path, and for cache coherent systems, geometrically increase traffic associated with cache/memory management.
- Programmer responsibility for synchronization constructs that insure "correct" access of global memory.
- Expense: it becomes increasingly difficult and expensive to design and produce shared memory machines with ever increasing numbers of processors.

1.3.2 Distributed Memory

- Like shared memory systems, distributed memory systems vary widely but share a common characteristic. Distributed memory systems require a communication network to connect inter-processor memory.

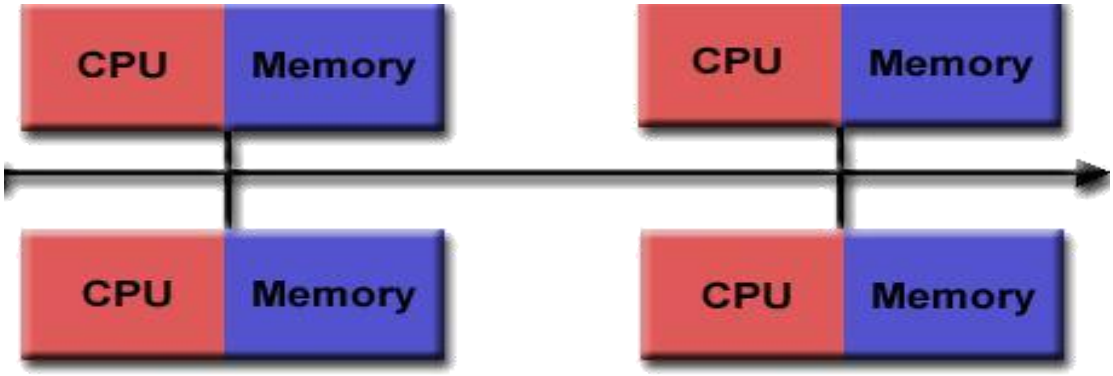


Figure distributed memory systems

- Processors have their own local memory. Memory addresses in one processor do not map to another processor, so there is no concept of global address space across all processors.
- Because each processor has its own local memory, it operates independently. Changes it makes to its local memory have no effect on the memory of other processors. Hence, the concept of cache coherency does not apply.
- When a processor needs access to data in another processor, it is usually the task of the programmer to explicitly define how and when data is communicated.

Synchronization between tasks is likewise the programmer's responsibility.

- Modern multicomputer use hardware routers to pass message. Based on the interconnection and routers and channel used the multicomputers are divided into generation

o 1st generation : based on board technology using hypercube architecture and software controlled message switching.

- 2nd Generation: implemented with mesh connected architecture, hardware message routing and software environment for medium distributed – grained computing.
- 3rd Generation : fine grained multicomputer like MIT J-Machine.

• The network "fabric" used for data transfer varies widely, though it can be as simple as Ethernet.

Advantages:

- Memory is scalable with number of processors. Increase the number of processors and the size of memory increases proportionately.
- Each processor can rapidly access its own memory without interference and without the overhead incurred with trying to maintain cache coherency.
- Cost effectiveness: can use commodity, off-the-shelf processors and networking.

Disadvantages:

- The programmer is responsible for many of the details associated with data communication between processors.
- It may be difficult to map existing data structures, based on global memory, to this memory organization.
- Non-uniform memory access (NUMA) times

1.2 The state of computing

Modern computers are equipped with powerful hardware technology at the same time loaded with sophisticated software packages. To access the art of computing we firstly review the history of computers then study the attributes used for analysis of performance of computers.

1.2.1 Evolution of computer system

Presently the technology involved in designing of its hardware components of computers and its overall architecture is changing very rapidly for example: processor clock rate increase about 20% a year, its logic capacity improve at about 30% in a year; memory speed at increase about 10% in a year and memory capacity at about 60% increase a year also the disk capacity increase at a 60% a year and so overall cost per bit improves about 25% a year.

But before we go further with design and organization issues of parallel computer architecture it is necessary to understand how computers had evolved. Initially, man used simple mechanical devices – abacus (about 500 BC) , knotted string, and the slide rule for computation. Early computing was entirely mechanical like : mechanical adder/subtracter (Pascal, 1642) difference engine design (Babbage, 1827) binary mechanical computer (Zuse, 1941) electromechanical decimal machine (Aiken, 1944). Some of these machines used the idea of a stored program a famous example of it is the Jacquard Loom and Babbage's Analytical Engine which is also often considered as the first real computer.

Mechanical and electromechanical machines have limited speed and reliability because of the many moving parts. Modern machines use electronics for most information transmission.

Computing is normally thought of as being divided into generations. Each successive generation is marked by sharp changes in hardware and software technologies. With

some exceptions, most of the advances introduced in one generation are carried through to later generations. We are currently in the fifth generation.

Ist generation of computers (1945-54)

The first generation computers where based on vacuum tube technology. The first large electronic computer was **ENIAC** (Electronic Numerical Integrator and Calculator), which used high speed vacuum tube technology and were designed primarily to calculate the trajectories of missiles. They used separate memory block for program and data. Later in 1946 John Von Neumann introduced the concept of stored program, in which data and program where stored in same memory block. Based on this concept **EDVAC** (Electronic Discrete Variable

Automatic Computer) was built in 1951. On this concept IAS (Institute of advance studies, Princeton) computer was built whose main characteristic was CPU consist of two units (Program flow control and execution unit).

In general key features of this generation of computers where

1) The switching device used where vacuum tube having switching time between 0.1 to 1 milliseconds.

2) One of major concern for computer manufacturer of this era was that each of the computer designs had a unique design. As each computer has unique design one cannot upgrade or replace one component with other computer. Programs that were written for one machine could not execute on another machine, even though other computer was also designed from the same company. This created a major concern for designers as there were no upward-compatible machines or computer architectures with multiple, differing implementations. And designers always tried to manufacture a new machine that should be upward compatible with the older machines.

3) Concept of specialized registers where introduced for example index registers were introduced in the Ferranti Mark I, concept of register that save the return-address instruction was introduced in UNIVAC I, also concept of immediate operands in IBM 704 and the detection of invalid operations in IBM 650 were introduced.

4) Punch card or paper tape were the devices used at that time for storing the program. By the end of the 1950s IBM 650 became one of popular computers of that time and it used the drum memory on which programs were loaded from punch card or paper tape. Some high-end machines also introduced the concept of core memory which was able to provide higher speeds. Also hard disks started becoming popular.

5) In the early 1950s as said earlier were design specific hence most of them were designed for some particular numerical processing tasks. Even many of them used decimal numbers as their base number system for designing instruction set. In such machine there were actually ten vacuum tubes per digit in each register.

6) Software used was machine level language and assembly language.

7) Mostly designed for scientific calculation and later some systems were developed for simple business systems.

8) Architecture features Vacuum tubes and relay memories CPU driven by a program counter (PC) and accumulator Machines had only fixed-point arithmetic

9) Software and Applications

Machine and assembly language Single user at a time

No subroutine linkage mechanisms Programmed I/O required continuous use of CPU

10) examples: ENIAC, Princeton IAS, IBM 701

IInd generation of computers (1954 – 64)

The transistors were invented by Bardeen, Brattain and Shockely in 1947 at Bell Labs and by the 1950s these transistors made an electronic revolution as the transistor is smaller, cheaper and dissipate less heat as compared to vacuum tube. Now the transistors were used instead of a vacuum tube to construct computers. Another major invention was invention of magnetic cores for storage. These cores where used to large

random access memories. These generation computers has better processing speed, larger memory capacity, smaller size as compared to pervious generation computer.

The key features of this generation computers were

1) The IInd generation computer were designed using Germanium transistor, this technology was much more reliable than vacuum tube technology.

2) Use of transistor technology reduced the switching time 1 to 10 microseconds thus provide overall speed up.

2) Magnetic cores were used main memory with capacity of 100 KB. Tapes and disk peripheral memory were used as secondary memory.

3) Introduction to computer concept of instruction sets so that same program can be executed on different systems.

4) High level languages, FORTRAN, COBOL, Algol, BATCH operating system.

5) Computers were now used for extensive business applications, engineering design, optimation using Linear programming, Scientific research

6) Binary number system very used.

7) Technology and Architecture

Discrete transistors and core memories I/O processors, multiplexed memory access Floating-point arithmetic available

Register Transfer Language (RTL) developed 8) Software and Applications

High-level languages (HLL): FORTRAN, COBOL, ALGOL with compilers and subroutine libraries Batch operating system was used although mostly single user at a time

9) Example : CDC 1604, UNIVAC LARC, IBM 7090

IIIrd Generation computers*(1965 to 1974)*

In 1950 and 1960 the discrete components (transistors, registers capacitors) were manufactured packaged in a separate containers. To design a computer these discrete unit were soldered or wired together on a circuit boards. Another revolution in computer designing came when in the 1960s, the Apollo guidance computer and Minuteman missile were able to develop an integrated circuit (commonly called ICs). These ICs made the circuit designing more economical and practical. The IC based computers are called third generation computers. As integrated circuits, consists of transistors, resistors, capacitors on single chip eliminating wired interconnection, the space required for the computer was greatly reduced. By the mid-1970s, the use of ICs in computers became very common. Price of transistors reduced very greatly. Now it

was possible to put all components required for designing a CPU on a single printed circuit board. This advancement of technology resulted in development of minicomputers, usually with 16-bit words size these system have a memory of

range of 4k to 64K.This began a new era of microelectronics where it could be possible design small identical chips (a thin wafer of silicon's). Each chip has many gates plus number of input output pins.

Key features of IIIrd Generation computers:

1) The use of silicon based ICs, led to major improvement of computer system. Switching speed of transistor went by a factor of 10 and size was reduced by a factor of 10, reliability increased by a factor of 10, power dissipation reduced by a factor of 10. This cumulative effect of this was the emergence of extremely powerful CPUS with the capacity of carrying out 1 million instruction per second.

2) The size of main memory reached about 4MB by improving the design of magnetic core memories also in hard disk of 100 MB become feasible.

3) On line system become feasible. In particular dynamic production control systems, airline reservation systems, interactive query systems, and real time closed lop process control systems were implemented.

4) Concept of Integrated database management systems were emerged.

5) 32 bit instruction formats

6) Time shared concept of operating system.

7) Technology and Architecture features

Integrated circuits (SSI/MSI) Microprogramming

Pipelining, cache memories, lookahead processing 8) Software and Applications

Multiprogramming and time-sharing operating systems Multi-user applications

9) Examples : IBM 360/370, CDC 6600, TI ASC, DEC PDP-82

IVth Generation computer (*(1975 to 1990)*

The microprocessor was invented as a single VLSI (Very large Scale Integrated circuit) chip CPU. Main Memory chips of 1MB plus memory addresses were introduced as single VLSI chip. The caches were invented and placed within the main memory and microprocessor. These VLSIs and VVSLIs greatly reduced the space required in a computer and increased significantly the computational speed.

1) Technology and Architecture feature semiconductor memory Multiprocessors,

vector supercomputers, multicomputers

Shared or distributed memory Vector processors

2) Software and Applications Multprocessor operating systems, languages,

compilers,

parallel software tools

Examples : VAX 9000, Cray X-MP, IBM 3090, BBN TC2000

Fifth Generation computers(1990 onwards)

In the mid-to-late 1980s, in order to further improve the performance of the system the designers start using a technique known as “instruction pipelining”. The idea is to break the program into small instructions and the processor works on these instructions in different stages of completion. For example, the processor while calculating the result of the current instruction also retrieves the operands for the next instruction. Based on this concept later superscalar processor were designed, here to execute multiple instructions in parallel we have multiple execution unit i.e., separate arithmetic-logic units (ALUs).

Now instead executing single instruction at a time, the system divide program into several independent instructions and now CPU will look for several similar instructions that are not dependent on each other, and execute them in parallel. The example of this design are VLIW and EPIC.

1) Technology and Architecture features ULSI/VHSIC processors, memory, and switches High-density packaging

Scalable architecture Vector processors

2) Software and Applications

Massively parallel processing Grand challenge applications Heterogenous processing

3) Examples : Fujitsu VPP500, Cray MPP, TMC CM-5, Intel Paragon

UNIT-2

PART-2

Data Representation:

1.2.1 Binary Numbers,

1.2.2Fixed Point Representation .

1.2.3Floating Point Representation.

1.2.4 Number base conversions

1.2.5 Octal and Hexadecimal Numbers,

1.2.6Signed binary numbers,

1.2.7Binary codes.

DATA REPRESENTATION

1.2.1. Binary Numbers:

The ancient Indian writer Pingala developed advanced mathematical concepts for describing prosody, and in doing so presented the first known description of a binary numeral system.A full set of 8 trigrams and 64 hexagrams, analogous to the 3-bit and 6-bit binary numerals, were known to the ancient Chinese in the classic text *I Ching*. An arrangement of the hexagrams of the *I Ching*, ordered according to the values of the corresponding binary numbers (from 0 to 63), and a method for generating thesame, was developed by the Chinese scholar and philosopher Shao Yong in the 11th century.

In 1854, British mathematician George Boole published a landmark paper detailing an algebraic system of logic that would become known as Boolean algebra. His logical calculus was to become instrumental in the design of digital electronic circuitry. In 1937, Claude Shannon produced his master's thesis at MIT that implemented Boolean algebra and binary arithmetic using electronic relays and switches for the first time in history. Entitled *A Symbolic Analysis of Relay and Switching Circuits*, Shannon's thesis essentially founded practical digital circuit design.

The radix (no of symbols in a number system is called radix) is 2. The positive binary numbers are stored in binary notation of sign magnitude representation and negative numbers are stored in sign magnitude or I's complement form or 2's complement form.

In I's complement and sign magnitude representation we have +0 and -0,

In 2's complement representation we have only one zero. Hence 2's complement representation is used in computers to store the data.

1.2.2. FIXED POINT REPRESENTATION:

It is used to represent integers either positive or negative. They are Sign Magnitude representation, I's complement representation and 2's complement representation

1.2.3 FLOATING POINT REPRESENTATION:

It is used to represent real numbers. It consists of mantissa which is fraction i.e. the first digit is a non zero digit for normalization condition and an exponent which is an integer. If the number of bits allocated for mantissa increases the accuracy of the number is increased, if the number of bits allocated for exponent increases the range of the number is increased.

1.2.4. NUMBER BASE CONVERSIONS

Any number in one base system can be converted into another base system Types

1) Decimal to any base

2) Any base to decimal

3) Any base to Any base

Decimal number: $123.45 = 1\ 10^2 + 2\ 10^1 + 3\ 10^0 + 4\ 10^{-1} + 5\ 10^{-2}$

Base *b* number: $N = a_{q-1}b^{q-1} + \ldots + a_0 b^0 + \ldots + a_{-p}b^{-p}$

$b > 1, \quad 0 <= a_i <= b-1$

Integer part: $a_{q-1}a_{q-2} \ldots a_0$

Fractional part: $a_{-1}a_{-2} \ldots a_{-p}$

Most significant digit: a_{q-1}

Least significant digit: a_{-p}

Binary number (*b*=2): $1101.01 = 1\ 2^3 + 1\ 2^2 + 0\ 2^1 + 1\ 2^0 + 0\ 2^{-1} + 1\ 2^{-2}$

Representing number *N* in base *b*: $(N)_b$

Decimal to Binary

Example: Convert $(432.354)_{10}$ to binary

Q_i	r_i
216	$0 = a_0$
108	$0 = a_1$
54	$0 = a_2$
27	$0 = a_3$
13	$1 = a_4$
6	$1 = a_5$
3	$0 = a_6$
1	$1 = a_7$
	$1 = a_8$

$0.354 \cdot 2 = 0.708$ hence $a_{-1} = 0$
$0.708 \cdot 2 = 1.416$ hence $a_{-2} = 1$
$0.416 \cdot 2 = 0.832$ hence $a_{-3} = 0$
$0.832 \cdot 2 = 1.664$ hence $a_{-4} = 1$
$0.664 \cdot 2 = 1.328$ hence $a_{-5} = 1$
$0.328 \cdot 2 = 0.656$ hence $a_{-6} = 0$
$a_{-7} = 1$
etc.

Thus, $(432.354)_{10} = (110110000.0101101\ldots)_2$

Octal To

B
i
n
a
r
y

Example: Convert $(123.4)_8$ to binary

$(123.4)_8 = (001\ 010\ 011.100)_2$

Example: Convert $(1010110.0101)_2$ to octal

$(1010110.0101)_2 = (001\ 010\ 110.010\ 100)_2 = (126.24)_8$

1.2.5. OCTAL AND HEXADECIMAL NUMBERS:

Octal is a 3-bit binary and Hexadecimal is a 4-bit binary.

1.2.7. SIGNED BINARY NUMBERS

Signed binary numbers are represented in three ways. They are

(a) Sign Magnitude representation
(b) I's complement representation
(c) 2's complement representation.

In first two representations zero is represented with two different binary numbers whereas in third i.e. 2's complement representation zero is represented with only one binary number. Hence this representation is used in computer for storing the signed binary numbers.

1.2.8. Binary codes

Binary codes are codes which are represented in binary system with modification from the original ones. Weighted Binary codes

Non Weighted Codes

Weighted binary codes are those which obey the positional weighting principles, each position of the number represents a specific weight. The binary counting sequence is an example.

Decimal	BCD 8421	Excess-3	84-2-1	2421	5211	Bi-Quinary 5043210			5	0	4	3	2	1	0
0	0000	0011	0000	0000	0000	0100001		0		X					X
1	0001	0100	0111	0001	0001	0100010		1		X				X	
2	0010	0101	0110	0010	0011	0100100		2		X			X		
3	0011	0110	0101	0011	0101	0101000		3		X		X			
4	0100	0111	0100	0100	0111	0110000		4		X	X				
5	0101	1000	1011	1011	1000	1000001		5	X						X
6	0110	1001	1010	1100	1010	1000010		6	X					X	
7	0111	1010	1001	1101	1100	1000100		7	X				X		
8	1000	1011	1000	1110	1110	1001000		8	X			X			
9	1001	1111	1111	1111	1111	1010000		9	X		X				

Reflective Code

A code is said to be reflective when code for 9 is complement for the code for 0, and so is for 8 and 1 codes, 7 and 2, 6 and 3, 5 and 4. Codes 2421, 5211, and excess-3 are reflective, whereas the 8421 code is not.

Sequential Codes

A code is said to be sequential when two subsequent codes, seen as numbers in binary

representation, differ by one. This greatly aids mathematical manipulation of data. The 8421 and Excess-3 codes are sequential, whereas the 2421 and 5211 codes are not.

Non weighted codes

Non weighted codes are codes that are not positionally weighted. That is, each position within the binary number is not assigned a fixed value. Ex: Excess-3 code

Excess-3 Code

Excess-3 is a non weighted code used to express decimal numbers. The code derives its name from the fact that each binary code is the corresponding 8421 code plus 0011(3).

Gray Code

The gray code belongs to a class of codes called minimum change codes, in which only one bit in the code changes when moving from one code to the next. The Gray code is non-weighted code, as the position of bit does not contain any weight. The gray code is a reflective digital code which has the special property that any two subsequent numbers codes differ by only one bit. This is also called a unit- distance code. In digital Gray code has got a special place.

Decimal Number	Binary Code	Gray Code	Decimal Number	Binary Code	Gray Code
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

Binary to Gray Conversion

- Gray Code MSB is binary code MSB.
- Gray Code MSB-1 is the XOR of binary code MSB and MSB-1.
- MSB-2 bit of gray code is XOR of MSB-1 and MSB-2 bit of binary code.
- MSB-N bit of gray code is XOR of MSB-N-1 and MSB-N bit of binary code