Computer Organization

UNIT- I

Prepared by
A.Srinivasan,
Assoc prof,
SITAMS,
chittoor

Computer Organization

Computer Organization refers to the Operational units and their interconnections that realizes the architectural specification.

Examples:

→ Control signals

→ Interfaces between computer and pheripherals

→ Memory technology being used.

Basic structure of Computers:

Computer types:

1. Digital computers
2. personel Computers
3. Notebook Computers
4. Workstations
5. Enterprise system
6. Server
7. Super computer.

Digital Computer:

→ It is a fast electronic calculating machine which accepts digitilized input information, process it to list of internally stored instruction and produces the resulting output information.

→ The list of instruction is called as computer program and internal storage is called computer memory.

→ The Computer may widely vary in size, cost and power.

Personel Computer:

→ The most Common computer is personel computer which can be used in homes, ~~schools~~, schools, business, office etc.

→ It is the most common form of desktop computer that have processing and storage unit, visual display, audio output unit and a keyboard.

→ In this storage media includes harddisc, CDROM and diskettes

Notebook Computers:

→ It is another version of personal computer where all the components are packed into a single unit

→ It is portable and size is of thin brief case.

Ex: Laptops, Tablet PC

Workstations:

→ Workstations are of with high resolution graphics and I/o capability that performs more Computational power than Personel Computer

→. It can be used in engineering application and interactive design work.

Enterprise System: (or) mainframes System:

→ It is a large and powerful System that are used for business data processing.

→ It computes at high power and has very large storage.

Servers:

→ It contains very large size database, storage unit and capable of handling large volumes of request to access the data

→ Servers are widely used in education, business and personel user Communities.

Super Computers:

→ It is used for large scale numerical Calculation required in applications such as weather forecasting, aircraft design and Simulation.

Functional units:

A computer consist of five functional units

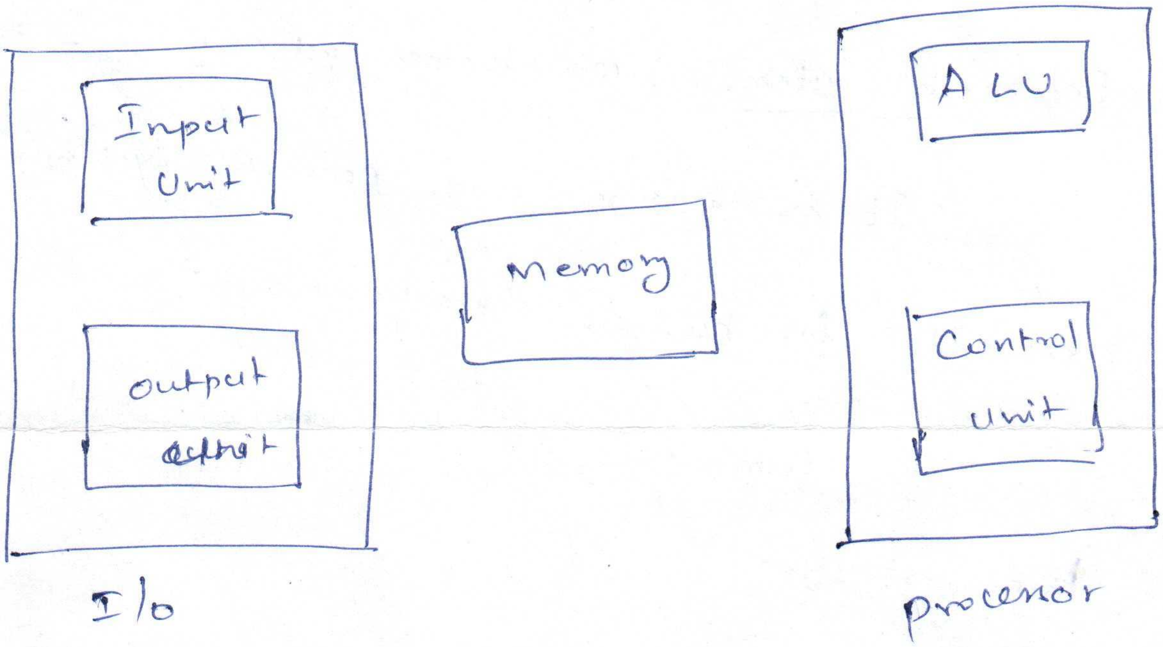
They are ① Input unit

② Memory unit

③ ALU

④ output unit

⑤ Control unit.



Basic functional units of Computer

→ The input unit accepts the coded information from the user through the electromechanical device such as keyboard. or from other computers.

→ The information received is either stored in memory or immediately used by ALU to perform desired operation.

→ The processing steps are determined by a program stored in the memory.

→ finally the results are sent back to the user through the output unit.

→ The input & output unit is often collectively called Input output unit.

→ The processor fetches the instruction from the memory and performs the desired operation.

→ The computer is completely controlled by the stored program, except for possible external interruption by an operator or by I/o devices Connected to machine.

→ Data are the numbers or encoded characters that are used as operands by the instruction.

→ As computer knows only machine language, the source program (HLL) is translated into machine language program by the compiler

→ Each number, character or instruction is encoded as a binary digits called bits, each have one of two possible values 0 or 1

→ Two standard binary code formats are ASCII (American standard code for Information Interchange) where each character is represented as a 7 bit code and EBCDIC (Extended binary coded decimal Interchange code) which is represented as a 8 bit Code.

Input unit:

→ Computers accepts coded information through Input units that reads the data.

→ The most well known input device is the keyboard where a key is pressed, the corresponding letter(or) digits is automatically translated into its corresponding binary code and the same is ~~stored or~~ transferred to either memory or the processor.

Memory unit:

→ The function of the memory unit is to store the programs and data

→ There are two classes of storage called Primary and Secondary Storage.

→ Primary memory is the fast memory where the programs are stored while they are executed.

→ The memory Contains a large number of Storage Cells, each capable of storing one bit of information

→ Each cells are used as a fixed size cell called as word. The memory is organized so that the Content of one word containing 'n' bits Can be stored or retrieved in one basic operation

→ Each cell in the memory is given a distinct address to provide easy access to the word.

→ The given word is accessed by specifying its address and issuing a Control Command Signal That starts Storing or retrieval process.

→ The no. of bits in each word is referred to word length. that ranges from 16 to 64 bits

→ Memory in which any location can be reached in short and fixed amount of time after Specifying its addres is called as RAM.

→ The time required to acces one word is called as memory access time.

→ Seconday storage is used to store large amount of data and many programs

→ The typical Seconday storage devices are magnetic disk, magnetic tapes, and optical disks.

Arithmetic and logic unit:

→ The basic aritametic operations are performed in ALU. Eg: two numbers to be added

~~They~~ → The two numbers are brought into the processor and actual addition is carried out by ALU

→ The sum may be stored in memory or retained in the processor for immediate use.

→ When operands are brought into processer they are stored in high speed storage element called registers.

→ Access time to register is faster than the access time to memory.

Output unit:

→ Its function is to send processors results to the outside world (or) user.

Ex: Monitor, printer etc.

→ Some units such as graphic displays provide both input function and output function.

Control unit:

→ The memory, ALU, i/p and o/p unit store and process information and perform i/o operation The Control unit sends the control signals to other units and senses their states.

→ The actual transfers are generated in the Control circuits by timing signals that determines when a given action is to take place.

The operation of Computer can be summarized as follows:

→ The computers accepts information in the form of programs and data through i/p unit and stores it in memory

→. ~~Information~~ Information stored in the memory is fetched under program control, into arithmetic & logic unit where it is processed.

→ Processed information leaves the computer through output unit

→. All the activities inside the computer are directed by the Control unit.

Basic Operational Concepts:

Consider the following instruction

Add LocA , Ro

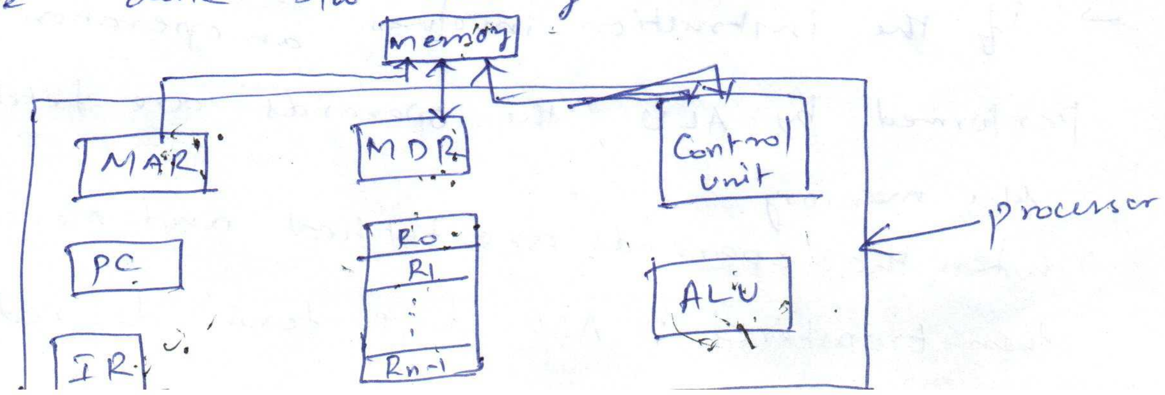
→ This instruction adds the operand at memory Loc to the operand in a register Ro, which is present in processor and their sum is stored in Ro which is overwritten

→ The above instruction performs the following steps

1. The instruction is fetched from the memory into processor.
2. The operand at LocA is fetched and added to the Contents of Ro
3. The resulting sum is stored in Register Ro

→ Transfer between memory and procuser are started by sending the address of memory location to be accessed to the memory unit and issuing appropriate Control signal

→ The below figures shows how the transfers can be done b/w memory and processors.



Instruction Register (IR):

It holds the instruction that is currently being executed

Program Counter (PC):

It contains the address of the next instruction that is to be fetched.

Memory address Register (MAR):

It holds the address which is to be accessed.

Memory Data register (MDR)

It contains the data to be written into or read out of the addressed location

→ Execution of the program starts when the pc is set to point to the first instruction of the program

→ The content of pc are transferred to the MAR and 'read' control signal is sent to the memory

→ Then the word is read out of the memory and loaded into the MDR

→ The contents of MDR are transferred to the IR.

→ If the instruction involves an operation to be performed by ALU, the operands are fetched from the memory.

→ when the operands are fetched and moved to MDR then transfered to ALU to perform desired operation.

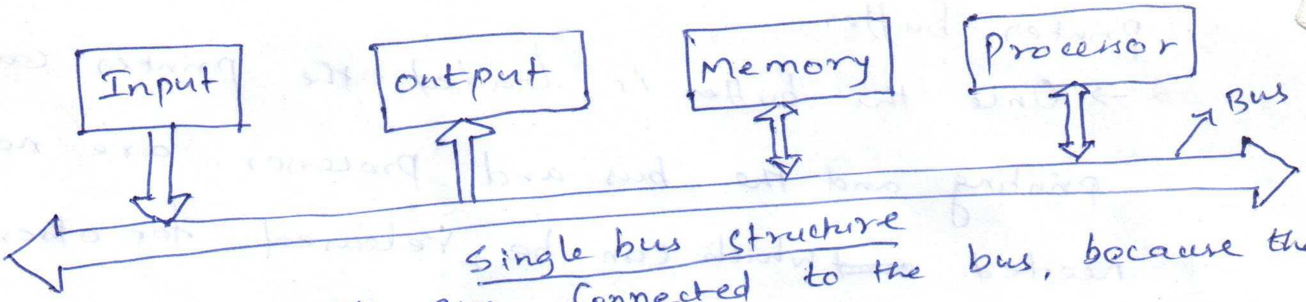
Bus structure :

→ when a word of data is transferred between units, all its bits are tratsfered in parallel i e the bits are transferred simoultaneously over many lines or wires, one bit per line.

→ A group of lines that serves as a connecting path for several devices is called as bus

→ The lines can carry the data, address and control signals

→ The simplest way to interconnect functional units is to use single bus which is shown below.



Single bus structure

→ All the units are connected to the bus, because the bus can be used for only one transfer at a time, only two units can actively use the bus at any given time

→ The main thing of single bus structure is its low cost and its flexibility for attaching pheripheral device

→ Systems that contains multiple buses achieve more concurrency in operations by allowing two or more transfers to be carried out at the same time. which leads to better performance but at an increased cost.

→ Memory and processors Operate at electronic speed making them the fastest part of a Computer.

→ All the devices must Communicate with each other over a bus, so an efficient transfer mechanism must be used to smooth out the differences in timing among processors, memories and external devices

→ A common approach is to include buffer registers with the devices to hold the information during transfer.

Ex:
→ Consider the tranfer of an encoded character from the processor to a character printer. The Processor Sends the character over the bus to the printer buffer.

→ Once the buffer is loaded, the printer can start printing and the bus and processor are no longer needed ~~and~~ which can be released for other activity.

→ The printer Continues printing the character in its buffer and it is not available for further transfer until this process is Completed.

→ Thus buffer registers Carry timing differences among processors, memory and I/o devices

Bus:

Bus Connects various Components in a Computer system

Bus types:

1. Internal bus (or) System bus
2. External bus
3. Control bus
4. Synchronous bus
5. Asynchronous bus
6. Data bus
7. address bus

Internal bus:

→ This bus is used inside the processor System

→ It is also refered as System bus, which is used to transfer data, address and Control signals

External bus:

→ This bus is used to interface with the devices outside the processor System.

→ It is typically used to Connect I/o devices.

Control bus:

→ The Control bus Carries the transaction Specific control information

→ Some typical Control Signals are

1. Memory Read and memory write
2. I/o Read and I/o write
3. Ready
4. Bus regnest and Bus grant
5. Interrupt and Interrupt acknowledgemnt.
6. DMA regnest and DMA acknowledgement
7. clock
8. Reset

Data bus:

→ A databus allows ~~the~~ Computer Subsystem for the transfer of data from one unit to other unit.

→. This includes transfering data to and from memory or from processor to other Components.

→ The amount of data a data bus can handle is Called as bandwidth.

→ A typical data bus is 32 bit wide means that 32 bit of data can travel through bus evry second.

→ Newer Computer data buses can handle 64 bit and even 96 bit data path.

Address bus:

→ A address is a Computer bus that is used to specify a physical address

→ when a processor needs to read or write to a memory location, it specifies that memry location on the address bus

→ The width of address bus is 32 bits

Synchronous bus:

→ A bus used to interconnect devices that Comprise a computer system where the timing of transactions between devices is under the Control of Synchronizing clock signal.

→ A device Connected to a synchronous bus must guarantee to respond to a command within the period set of the clock signal or transmission error will occur

Asynchronous bus:

→ A bus used to interconnect devices ~~that~~ b/w of the Computer system where information transfer devices are self timed rather than Controlled by synchronising clock signal.

→ A Connected device indicates its readiness for transfer by activating a request signal

→ A responding device indicates the Completion of transfer by activating an acknowledgement signal.

Software: → Computer s/w is the list of instruction that makes the hardware friendly to the user.

→ System Software is a collection of programs that are executed as needed to perform functions such as.

① Receiving and interpreting user commands

② Entering, editing application Program and storing them as files in secondary storage device.

③ Managing the storage and retrieval of files in secondary storage device

④ Running the standard application programs such as word processor, Spread Sheets or games with data supplied by the user

⑤ Controlling I/O units to receive input information and produce output result.

⑥ Translating the program from source program to the object form

⑦ Linking and running user written application programs with existing standard library functions

→ Examples of system Software are Compiler, operating System, Interpreter, editors and so on

→ Application s/w are developed to assist the user to perform specific task.

→ In order to run an application program, the Computer must already contain these kinds of system softwares.

Performance:

→ The most important measure of the computer is its performance ie how quickly it can execute the programs.

→ Let us see, some of the performance measures

① processor clock:

→ Processor Circuits are Controlled by a timing signals called as a clock, ~~that~~ The clock defines regular time intervals called clock cycle

→ To execute a machine instruction, the processor divides the action to be performed into a sequence of basic step; such that each step can be completed in one clock cycle

② Basic performance equation:

Let T be the processors time required to execute a program that has been prepared in some high level language.

∴ The processor execution time for a program is given by $T = \frac{N \times S}{R}$.

where N = No. of actual machine instruction

S = avg No. of steps to be taken to execute one machine instruction.

R = clock rate per second.

Pipelining and superscalar operation :

→ The improvement of a performance can be ~~executed~~ achieved by overlapping the execution of successive instruction called Pipelining.

→ When multiple no. of instructions are to be executed by creating parallel paths ie the several instruction can be executed in every clock cycle. This mode of operation is called superscalar operation

clock rate:

There are two possibilities for increasing the clock rate. They are

① Improving the integrated circuit technology to work in a fast way

② Reducing the amount of processing done in one basic step.

Instruction set : CISC and RISC :

→ Complex instruction set computer has bigger instruction set ie, it involves more number of steps for processing a single instruction

→ Reduced instruction set computer has smaller instruction set ie it requires minimum number of steps for processing

Compiler:

→. A compiler translates a high level language program into sequence of machine instructions. So we need to ~~provi~~. have a suitable machine instruction set, then the compiler will provide good performance.

Performance Measurement:

→ The Computer Community adopted the idea of measuring Computer paerformance using benchmark programs.

→. A non profit organization called System Performance Evaluation Corporation (SPEC) selects and publishes the representative application programs for different domains.

Multiprocessors and multicomputers:

→ Large Computer systems may contain number of processors units, in which they are called as multiprocessors system.

→ There system execute a number of different task in parallel or they execute subtasks of a single large task in parallel.

→ All the processors have access to all of the memory in such systems. and they are termed as shared memory multiprocessor system.

→ The high performance of these system comes with much increased complexity and cost.

→ In addition to multiple processors and memory unit the cost is increased because of the need for more complex interconnection Networks

→ In Contrast, it is also possible to use an interconnected group of Complete Computers to achieve high total Computational power.

→ when the tasks they are executing need to communicate data, they do so by exchanging messages over a Communication Network. This is referred as message passing multicomputers.

Data Representation:

Datatypes:

→ The datatypes found in the registers of digital Computers may be classified as

(1) Numbers used in arithmetic Computation

(2) Letters of the alphabet used in data processing

(3) Other discrete symbols used for specific purposes

→ we represent each data in binary form in the registers

Number System:

radix:

→ A number System of base or radix r is a System that uses distinct symbol for r digits. To determine the quantity ~~of~~ the number ~~represents~~, It is necesary to multiply each digit by an integer power of r and then form the sum of all weighted digits.

Decimal:

→ The Decimal system number uses the radix 10 system.

→ The 10 Symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

→ The digit 824.5 is represented as.

$$8 \times 10^2 + 2 \times 10^1 + 4 \times 10^0 + 5 \times 10^{-1}$$

which indicates that 8 hundreds plus 2 tens plus 4 units plus 5 tenths.

Binary:

→ The binary number System uses the radix 2. The two digits will be 0 and 1.

→ The string of digits 101101 is interpreted as

$$1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0=45$$

→ To distinguish between different radix numbers the digit will be enclosed in parenthesis and the radix of the number inserted as a subscript.

Ex: $(101101)_2=(45)_{10}$

octal & hexadecimal:

→ The octal (radix 8) and hexadecimal (radix16) The eight symbols are 0, 1, 2, 3, 4, 5, 6, 7. The sixteen symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

→ The symbols, A, B, C, D, E, F Corresponds to the decimal numbers 10, 11, 12, 13, 14, 15 respectively.

→ The number in radix 'r' can be converted to the familar decimal system by forming the sum of the weighted digits.

Eg: octal 736.4 to decimal

$$(736.4)_8 = 7\times8^2+3\times8^1+6\times8^0+4\times8^{-1}$$

$$= 7\times64+3\times8+6\times1+\frac{4}{8} = (478.5)_{10}$$

The decimal number of hexadecimal F3 is

$$(F3)_{16} = F\times16^1+3\times16^0 = (15\times16)+3 = (243)_{10}$$

Convertion:

→ The Conversion of decimal 41.6875 into binary is done by first seperating the number into its integral part 41 and fraction part .6875.

→ The integral part is converted by dividing 41 by r=2 to an integer quotient 20 and a remaindes 1.

→ The quotient is again divided by 2 to give a new quotient and remainder.

→ This process is repeated until the integer quotient becomes 0.

→ The first remainder will give the low-order bit of the converted binary number.

```
2 | 41
2 | 20 - 1
2 | 10 - 0
2 | 5 - 0
2 | 2 - 1
2 | 1 - 0
    0 - 1
```

$(41)_{10} = (101001)_2$

→ The fraction part is converted by multiplying it by 2 to give an integer and a fraction.

→ The new fraction is again multiplied by 2 to give a new integer and a new fraction.

→ This process is repeated until fraction part becomes zero

→ finally two parts are combined to give total required conversion.

Integral part

```
2 | 41
2 | 20 - 1   ↑
2 | 10 - 0
2 |  5 - 0
2 |  2 - 1
     1 - 0
```

$(41)_{10} = (101001)_2$

fraction part

0.6875

0.6875 × 2

1.3750 × 2

0.7500 × 2

1.5000 × 2

1.0000

$(0.6875)_{10} = 0.1011$

$\therefore (41.6875)_{10} = (101001.1011)_2$

Octal and hexadecimal numbers:

→ The conversion from and to binary, octal & hexadecimal plays an important role in digital computer.

→ Each octal digits represents to 3 and each hexadecimal digits represents to 4 binary digits.

→ conversion from binary to hexadecimal is from four digits.

Ex:-

```
   A     F     6     3      Hexadecimal
 1010  1111  0010  0011     Binary
 1  2    7   5   4   3      octal.
```

Octal Number	Binary Code	Decimal equivalent
0	000	0
1	001	1
2	010	2
3	011	3
4	100	4
5	101	5
6	110	6
7	111	7

Hexadecimal Number	Binary Code	Decimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
:		
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15
14	0001 0100	20
F8	1111 1000	248

Decimal representation:

→ The binary number System is the most natural system for a Computer, but people are experienced to the decimal System.

→ One way to solve this Conflict is to Convert all i/p decimal numbers into a binary number

→ The another alternative bit assignment most commonly used for the decimal digits is the straight binary assignment listed in the following table is known as BCD.

Decimal	BCD
0	0000
1	0001
2	0010
3	0011
:	
10	0001 0000
248	0010 0100 1000

→ The only difference b/w a binary number and BCD is that in BCD each decimal digit is represented seperately by a four bits.

Complements:

→ Complements are used in digital Computers for simplying the subtraction operation and for logical manipulation

→ There are two types of Complements ① r's complement.
② (r-1)'s complement

When the values of base r is substituted, the two types are reffered
→ ~~The [illegible] complement are~~ 1's and 2's Complement for binary numbers and ~~[illegible]~~ 10's & 9's Complement for decimal numbers.

(r-1)'s Complement:

9's complement Given a number N in base r having 'n' digits, the (r-1)'s Complement of N is defined as $(r^n-1)-N$.

Eg:- $r=10$ $\therefore r-1=9$

So the 9's Complement is $(10^n-1)-N$. Now 10^n represents a number that Consist of a single 1 followed by n 0's. 10^n-1 is number represented by n 9's

eg: with $n=4$ we have $10^4-1 = 9999$

→ The 9's Complement is then obtained by subtracting each digit from 9.

Ex:. 9's Complement of 546700, 12389

$999999 - 546700 = 453299$

$99999 - 12389 = 87610$

1's complement:

for binary numbers $r=2$ & $r-1=1$ so the 1's complement of N is $(2^n-1)-N$

Ex: $n=4$, $(2^4-1) = 15 = (1111)_2$

The 1's complement of a binary number is formed by changing 1's into 0's and 0's into 1's.

Ex:. The 1's Complement of 1011001 is 0100110

→ The (r−1)'s Complement of octal or hexadecimal numbers are obtained by subtracting each digit from 7 and F (decimal 15) respectively.

r's Complement:

10's complement:

→ The r's Complement of a 'n' digit number N in base r is defined as $r^n - N$ for $N \neq 0$ and 0 for $N=0$.

→ Thus the 10's Complement of the decimal 2389 is $(10000 - 2389) = 7611$

→ r's Complement is obtained by adding 1 to (r−1)'s complement since $r^n - N = [(r^n-1)-N]+1$

The 2's complement of binary 101100 is

010011 + 1 = 010100

Subtraction of unsigned numbers:

→ The direct method of subtraction uses the borrowing concept

→ The subtraction of two n digit unsigned numbers $M-N$ $(N \neq 0)$ in base r can be done as follows

1. Add the minuend M to the r's complement of the subtrahend N. This performs $M+(r^n-N) = M-N+r^n$

2. If $M \geq N$ the sum will produce an end carry r^n which is discarded and what is left is the result $M-N$

3. If $M<N$, the sum doesn't produce an end carry and is equal to $r^n-(N-M)$, which is the r's complement of $(N-M)$

Ex: 72532 − 13250 = 59282

The 10's complement of 13250 is

99999 − 13250 = 86749 + 1 = 86750

$$M = 72532$$

10's complement of N = 86750

$$\text{Sum} = \boxed{1}59282$$

Discard the end carry 1 and Answer = 59282

Ex M N   ie M = 13250

N = 72532

produces negative 59282.

M = 13250

10's complent of N = 27468

Sum = 40718

There is no end carry

∴ Answer is negative 59282 which is 10's complement of 40718

10's Complent of 72532

99999 − 72532

= 27467 + 1

= 27468

10's complent of 40718

∴ 99999

(−) 40718

59281

(+) 1

59282

Binary Numbers:

① X = 1010100   Y = 1000011

X = 1010100

2's complemt of Y = 0111101

$\boxed{1}$ 0010001

Discard the end Carry

∴ Answer is 0010001

2's complemt of 1000011

0111100

+ 1

0111101

② ~~X~~ = 1000011

Y = 1010100   ∴ X = 1000011

2's complemt Y = 0101100

There is no end carry   1101111

∴ Answer is negative 0010001 which is 2's complement of 1101111

2's complemt of 1010100

0101011

+ 1

0101100

2's complent of 1101111

0010000

1

0010001

Fixed Point representation:

→ There are two ways , we are going to represent the Position q the binary point in a register.

① fixed Position

② floating point representation

→ Fixed point method assumes that the binary point is always fixed in one position.

→ There are two positions most widely used. They are

① A binary Point in the extreme left of the register to make the stored number a fractio

② A binary point in the extreme right of the register to make the stored number as an integer.

Integer representation:

→ when an integer binary number is Positive the sign is repremted by 0 and the magnitude by a Positive binary number.

→ when an integer binary number is negative the sign is represented by 1, but the rest of the number may be represented in one of three Possible ways.

1. Signed magnitude representation (SMR)

2 Signed 1's Complement "

3. Signed 2's Complement "

→ The signed magnitude represenftation of a negative numbe consist of the magnitude and a negative sign. Remaining 2, we are going to periform either 1's (or) 2's complemnt

Ex: +14 = 0000 1110 — 8 bit register.

Leftmost bits are zero in 8 bit register.

−14 can be reprented in 3 ways.

→ The signed magnitude represntation of −14 is obtained from +14 by Complementing only the sign bit.

Ex: −14 = 1000 1110 — (SMR)

→. The signed 1's complement represntation of −14 is obtained by Complementiy all the bits of +14, including the sign bit

Ex: −14 → 1111 0001 (Signed 1's Complmt

→ The signed 2's Complemt is obtained by takein the 2's Complement of positive number including its sign bit

Ex- −14 = 1111 0001 = 1111 0010

(+) 1

1111 0010 (Signed 2's complement,

Arithmetic addition:

→ The addition of two numbers in the Signed Magnitude representation follows the rules. of ordinary arithmetic.

→ If the sign are the same we add the two magnitudes and give the sum the common sign

→ If the signs are different, we subtract the smaller magnitude from the larger magnitude and give result the sign of the larger.

+6	00000110		−6 −	11111010 (2's comp)
+13	00001101		+13 −	00001101
+19	00010011		+7	1) 00000111

→ In Contrast 2' complement addition is stated as follows.

→ Add the two numbers, including their sign bits and discard the carry out of the sign bit position.

→ The negative numbers must initially be in 2's complement and that if the sum obtained after the addition is negative, it is in 2's complement form.

+6	00000110		−6	11111010 (2's)
−13	11110011 (2's complmt)		−13	11110011 (2's)
−7	11111001 (2's complmt)		−19	11101101
				(2's comple)

Arithmetic Subtraction :

→ Subtraction of two signed binary numbers when negative numbers are in 2's Complement form is very simple and can be stated as follows:

→. Take the 2's complement of the subtrahend (including sign bit) and add it to the minuend (including sign bit). A carry out of sign bit is discarded

→ A subtraction operation can be changed to an addition operation if the sign of the subtrahend is changed

→ This can be demonstrated through the relationship.

$$(\pm A) - (+B) = (\pm A) + (-B)$$

$$(\pm A) - (-B) = (\pm A) + (+B)$$

→ But changing a positive number to a negative number is easily done by taking its 2's complement. The reverse is also true.

Eg: $(-6) - (-13) = +7$

11111010 − 11110011

The subtraction is changed to addition by taking the 2's complement of the subtrahend (−13) to give (+1

ie $11111010 + 00001101 = \boxed{1}00000111$

By removing the end carry we obtain the correct answer 00000111(+7

Overflow:

→ when two numbers of n digits are added and the sum occupies n+1 digits, we say that an overflow occured.

→. An overflow is a problem in digital Computer because width of the registers is finite.

→ The detection of an overflow after addition of two binary numbers depends on sign of the number ie whether it is signed or unsigned number.

→ when two unsigned numbers are added, an overflow is detected from the end carry..

→ In case of signed number the left most bit always represents the sign and negative numbers are in 2's complement form.

→ when two signed numbers are added, the sign bit is treated as part of number and end carry doesn't indicate an overflow.

→ An overflow doesn't occur if one number is positive and another is negative

```
                                          Carries: 1 0
      Carries :  0 1                       -70 - 10111010
Eg:    +70  -  01000110                    -80 - 10110000
       +80  -  01010000                          ---------
       ----    --------                   -150   01001010
      +150     10010110                   ----   ---------
      ----    ---------
```

→ Note that 8 bit result should have been positive but has a negative sign bit in first Calulation and 8 bit result should have been negative but [illegible] in second Calculation

→ howeven the Carry out of sign bit position is taken as sign bit of the result, the 9 bit answer so obtained will be correct.

→ Since the answer cannot be accomodated within 8 bits we say that an overflow occured.

Overflow detection:

→ The overflow can be detected by observing the the carry into the sign bit and carry out of the sign bit position.

→ If there two carries are not equal, an overflow condition produced.

→ If these two carries are applied to exclusive OR gate, an overflow will be detected, when o/p of the gate is equal to '1'

Decimal fixed point representation:

→ The representation of decimal ~~bo~~ number in a register is a function of the binary code used to repreent a decimal digit.

→ A 4 bit decimal code requires four flipflops for each decimal digit. in binary code.

→ The representation of 4385 in BCD requires 16 flipflops ie four flipflops for each digits

0100 0011 1000 0101

→ By representing number in decimal we are wasting a considerable amount of Storage Space Since number bits required to store a decimal number in binary code is greater than the number of bits needed for its equivalent binary representation.

→ However the advantage in using decimal representation is computer i/p and o/p data are generated by people who use only the decimal system

→

Floating point representation:

→ The floating point representation of a number has two parts

① first part represents a signed fixed point called mantissa

② The second part designates the position of the decimal point and is called the exponent.

→ The fixed point mantissa may be a fraction or an integer

Eg:- The decimal number +6132.789

fraction	Exponent
+0.6132789	+04

→ The value of the exponent indicates that the actual position of the decimal point is four position right of the indicated decimal point in the fraction

→ The equivalent value is represented as $+0.6132789 \times 10^{+}$

→ The floating point is represented as

$$m \times r^{e}$$

Only the ~~mantissa~~ mantissa m and the exponent e are physically represented in the register (including sign)

→ A floating point binary number is represented in a similar manner except that it used base 2.

Eg: Binary number +1001.11 is represented with 8 bit fraction and 6 bit exponent as follows

fraction	exponent
01001110	000100

→ The fraction has a 0 in the leftmost position to denote Positive. The decimal point of the fraction follows the sign bit but is not shown in the register

→ The exponent has the equivalent binary number for +4

→ The floating point number is equivalent to

$$m \times 2^{e} = +(.1001110)_2 \times 2^{+4}$$

Normalization :

→ A floating point number is said to be normalized if the MSB of the mantissa is non zero

Ex: The decimal number 350 is normalized but 00035 cannot be normalized.

→ Regardless of where the position of the radix point is assumed to be in the mantissa, the number is normalized only if its leftmost digit is nonzero.

Eg: 8 bit binary number 00011010 is not normalized because the MSB is zero.

→ The floating point numbers are more Complicated than the fixed point numbers and their execution takes longer time and requires more Complex H/w.

Error Detection Codes:

→ Binary information are transmitted through some Communication medium which is subject to external noise that could change bits from 1 to 0 and vice versa.

→ In most Practical System, there is always a finite probability of occurance of a errors

→ Two types of Codes like error detection and error correction codes are used to detect and correct the errors

→ An error detection code is a binary code that detects digital errors during transmission. The detected errors cannot be corrected but their presence is indicated.

→ An error Correcting Code is a binary Code that detects and corrects the bit errors occured during transmission

Parity bit:

→ The most Common error detection code used is the parity bit.

→ A parity bit is an extra bit included with the binary message to make the total number of 1's either odd or even

→ The message of 3 bits and two possible parity bit is shown below

Message x y z	p(odd)	p(even)
000	1	0
001	0	1
010	0	1
011	1	0
100	0	1
101	1	0
110	1	0
111	0	1

→ p(odd) bit is chooren in such a way as to make sum of 1's (in all four bits) odd.

→ P(even) bit is choosen in such a way as to make sum of 1's even.

→ The even parity scheme has the disadvantage of having bit combination of all 0's while in the odd parity there is always one bit ie 1.

→ The p(odd) is a complement of p(even).

Parity generator:

→ During the transfer of information from one location to another, the parity bit is handled as follows.

→ At the sending end the message (in case 3 bits) is applied a parity generator, where the required parity bit is generated.

→ The message including the parity bit is transmitted to its destination.

Parity checker:

→ At the receiving end all the incoming bits are applied to parity checker that checks the proper parity bit ~~or~~ is transmitted to its destination

→ An error is detected if the checked parity does not conform to the adopted parity.

→ The parity methods detects the presence of one, three or any odd numbers of errors. An even number of errors is not detected.

odd functions:

→ ① Parity generator and checker network are logic circuits constructed with XOR and XNOR gates

→ ② An odd function is a logic function whose binary value is 1 iff an odd no. of variable are equal to 1

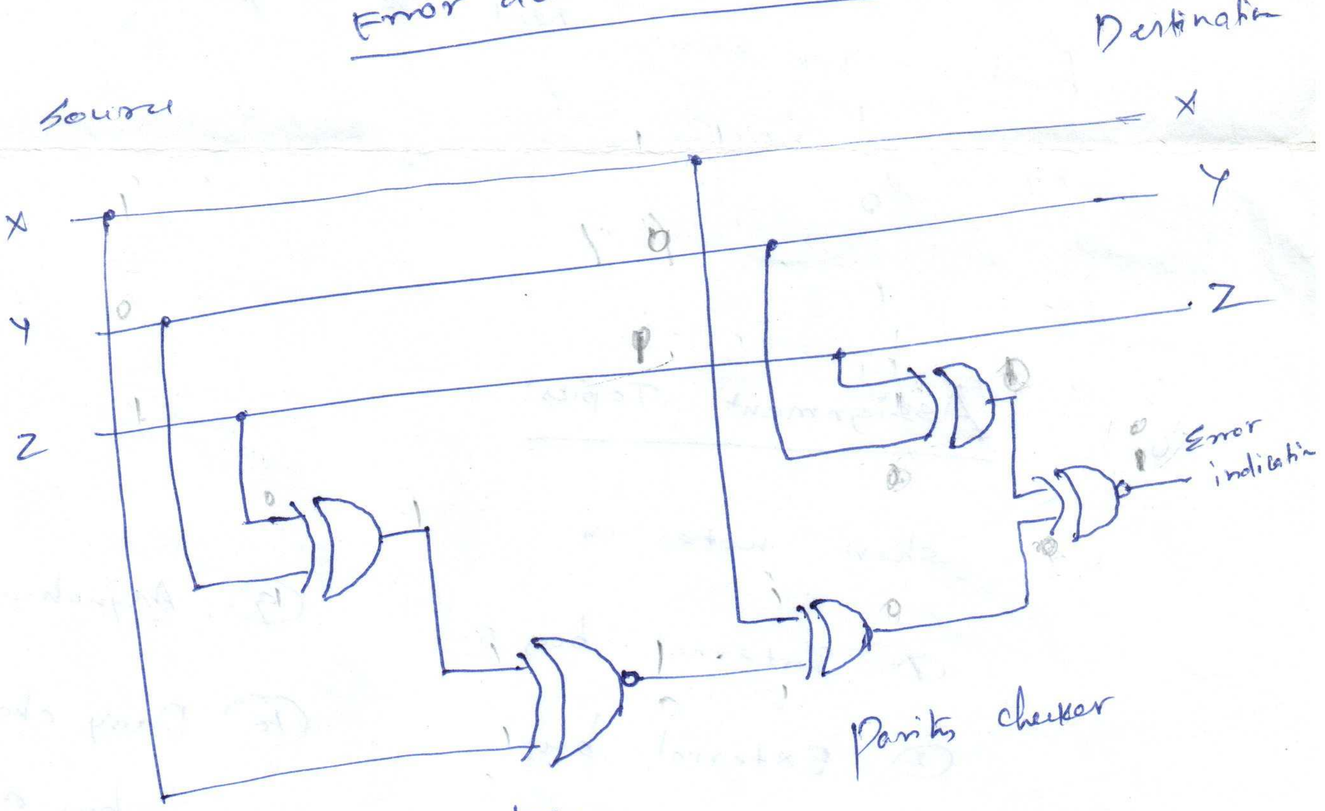
p(odd) is the complement of p(even), XNOR gate is used to complement. ~~to~~

② → According the this definition P(even) function is the XOR of X, Y, Z, because it is equal to 1 when either one or all three of the variable are equal to 1

Example:

A 3 bit message to be transmitted with an odd parity bit. At the sending end, the odd parity bit is genrated by a parity generator circuit which is shown below

Error detection with odd parity



Parity Generator.

→ This circuit Consist of one XOR and one X-NOR gate since P(even) is the ~~[illegible]~~ XOR of XYZ and P(odd) is the Complement of P(even)

→ The message and the odd parity bits are transmitte to their destination where they are applied to the parity checker.

→ An error has occured during transmission if the parity of the four bits received is even. since binary information transmitted was originally odd.

→ The o/p of the parity checker would be 1 when an error occurs that is when number of 1's in the four i/p is even.

→ Since XOR function of 4 input is a odd function we again need to complement the o/p by using XNOR gate.

Assignment Topics:

Write short notes on

1. Internal bus
2. External bus
3. Back plain bus
4. I/o bus
5. System bus
6. Address bus
7. Data bus
8. Synchronous bus
9. Asynchronous bus
10. Daisy chained based bus arbitration.

Important questions:

① a) What are the functional units? Explain each one

b) Explain about different buss in a practical Computer System. ~~and~~

② a) What are the different Performance measures used to represent a Computer system performance.

b) What do you mean by a parity bit? Explain with example how even and odd parity bits are generated. Is it possible to correct the errors using parity bits.

③ a) Explain about sign magnitude and 2's Complement approaches for representing the fixed point numbers.

b) Design a 4 bit odd parity generator and checker. Can parity bit be used for error detection. if so how?

④ a) Explain about various buses such as internal, external, backplain, I/o, system, address, data, Synchronous and asynchronous

b) Explain about daisy chain bus arbitration.