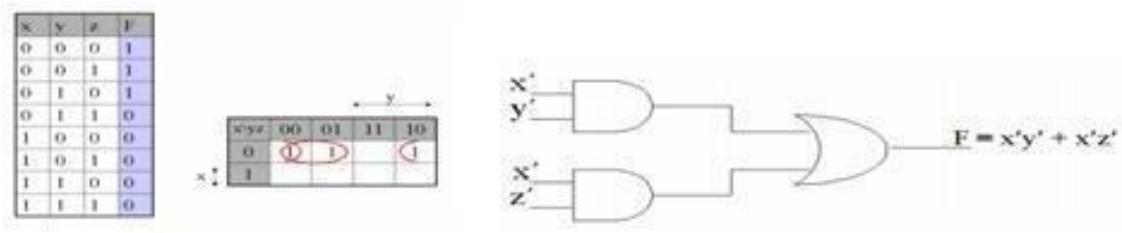
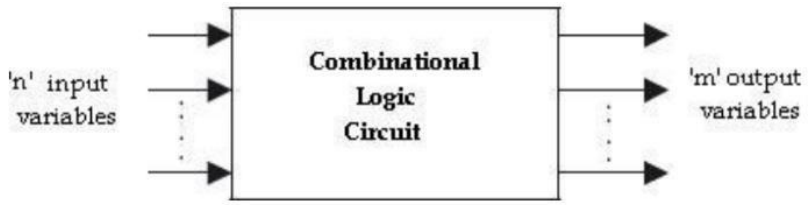
UNIT I COMBINATIONAL LOGIC

1. Design the combinational circuit with 3 inputs and 1 output. The output is 1 when the binary value of the input is less than 3. The output is 0 otherwise. (May/June 2016)



2. Define Combinational circuit? **(May/June 2016)**

Combinational circuit consists of logic gates whose output at any time is determined from the present combination of inputs. The logic gate is the most basic building block of combinational logic. The logical function performed by a combinational circuit is fully defined by a set of Boolean expressions.



Block diagram of a combinational logic circuit

3. Define Sequential logic circuit?

Sequential logic circuit comprises both logic gates and the state of storage elements such as flip-flops. As a consequence, the output of a sequential circuit depends not only on present value of inputs but also on the past state of inputs.

4. Write the Design Procedure.

Any combinational circuit can be designed by the following steps of design procedure.

1. The problem is stated.

2. Identify the input and output variables.

3. The input and output variables are assigned letter symbols.

4. Construction of a truth table to meet input -output requirements.

5. Writing Boolean expressions for various output variables in terms of input variables.

6. The simplified Boolean expression is obtained by any method of minimization—algebraic method, Karnaugh map method, or tabulation method.

7. A logic diagram is realized from the simplified boolean expression using logic gates.

5. What are the guidelines should be followed while choosing the preferred form for hardware implementation?

1. The implementation should have the minimum number of gates, with the gates used having the minimum number of inputs.

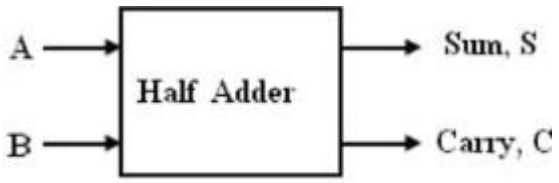
2. There should be a minimum number of interconnections.

3. Limitation on the driving capability of the gates should not be ignored.

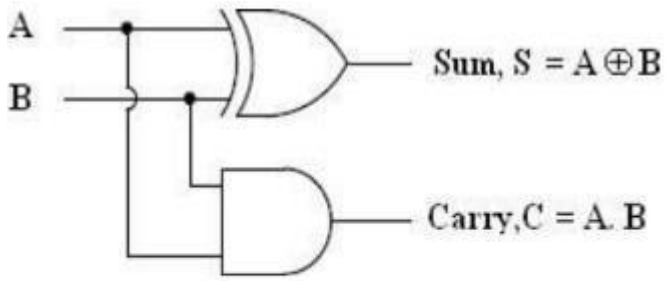
6. Explain half-adder with diagram.

A half-adder is a combinational circuit that can be used to add two binary bits. It has two inputs that represent the two bits to be added

and two outputs, with one producing the SUM output and the other producing the CARRY.



Block schematic of half-adder



Logic Implementation of Half-adder

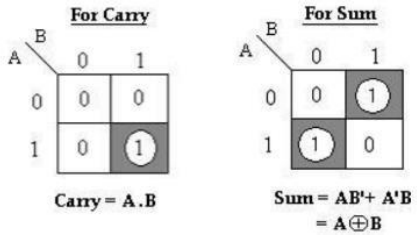
7. Draw the truth table of a half-adder. **(Nov./Dec. 2015)**

Inputs		Outputs	
A	B	Carry (C)	Sum (S)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Truth table of half-adder

8. Draw the K-map simplification for half adder.

K-map simplification for carry and sum:



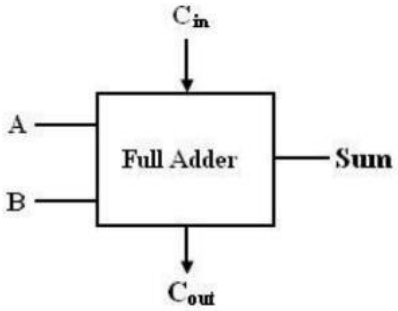
The Boolean expressions for the SUM and CARRY outputs are given by the equations,

Sum, S = A'B+ AB'= A†B

Carry, C = A . B

9. Define full adder.

A full adder is a combinational circuit that forms the arithmetic sum of three input bits. It consists of 3 inputs and 2 outputs. Two of the input variables, represent the significant bits to be added. The third input represents the carry from previous lower significant position. The block diagram of full adder is given by,

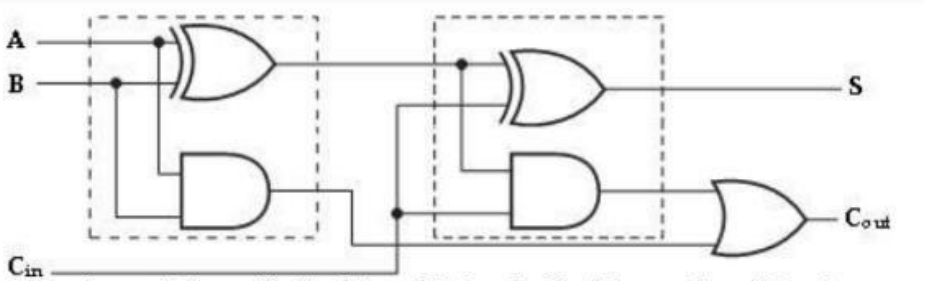


Block schematic of full-adder

10. Draw the truth table of a full-adder.

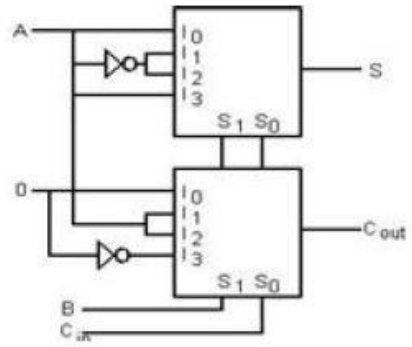
Inputs			Outputs	
A	B	C_{in}	Sum (S)	Carry (C_{out})
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

11. Draw the Implementation of full adder with two half-adders and an OR gate.

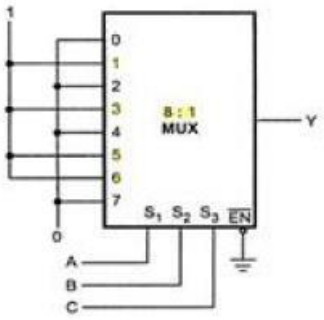


Implementation of full adder with two half-adders and an OR gate

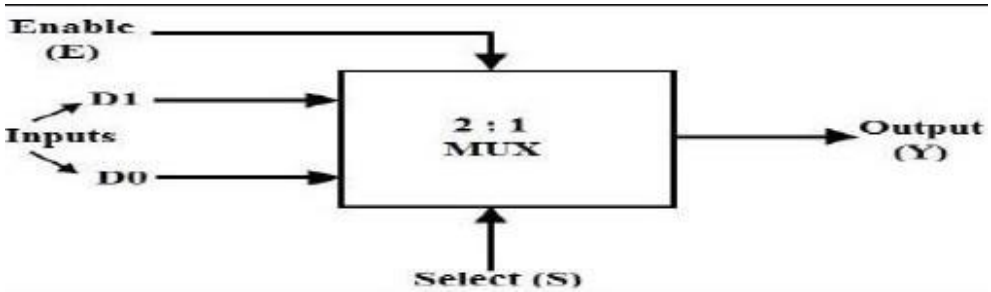
12. Implement a full adder with 4×1 multiplexer. (May 2015)



13. Implement the following Boolean function using 8:1 multiplexer F(A,B,C)= Σm(1,3,5,6)(Dec 2014)

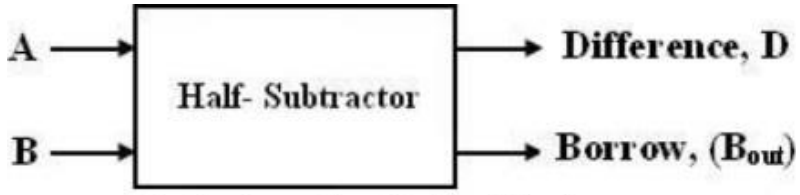


14. Draw a 2 to 1 multiplexer circuit. (June 2014)



15. Define half-subtractor.

A half-subtractor is a combinational circuit that can be used to subtract one binary digit from another to produce a DIFFERENCE output and a BORROW output. The BORROW output here specifies whether a ‗1' has been borrowed to perform the subtraction.

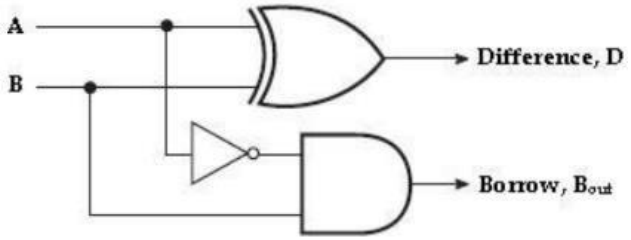


Block schematic of half-subtractor

16. Draw the truth table of half-subtractor.

Input		Output	
A	B	Difference (D)	Borrow (B_{out})
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

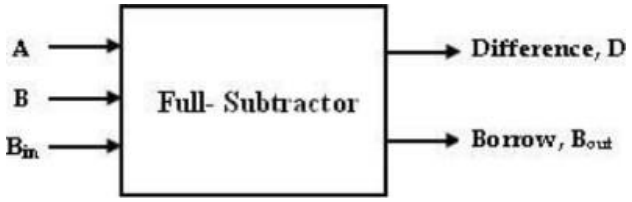
17. Draw the logic diagram of the half subtractor.



Logic Implementation of Half-Subtractor

18. Define full subtractor.

A full subtractor performs subtraction operation on two bits, a minuend and a subtrahend, and also takes into consideration whether a 1' has already been borrowed by the previous adjacent lower minuend bit or not. As a result, there are three bits to be handled at the input of a full subtractor, namely the two bits to be subtracted and a borrow bit designated as Bin. There are two outputs, namely the DIFFERENCE output D and the BORROW output Bo.

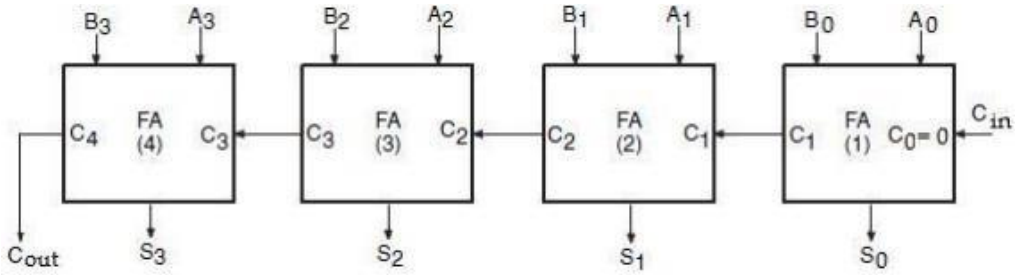


19. Draw the truth table for full-subtractor.

Inputs			Outputs	
A	B	B_{in}	Difference(D)	Borrow(B_{out})
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

20. Define Binary Adder (Parallel Adder).

The 4-bit binary adder using full adder circuits is capable of adding two 4- bit numbers resulting in a 4-bit sum and a carry output.



4-bit binary parallel Adder

Since all the bits of augend and addend are fed into the adder circuits simultaneously and the additions in each position are taking place at the same time, this circuit is known as parallel adder.

21. Define ripple-carry adder.

The bits are added with full adders, starting from the least significant position, to form the sum it and carry bit. The input carry C0 in the least significant position must be 0. The carry output of the lower order stage is connected to the carry input of the next higher order stage. Hence this type of adder is called ripple-carry adder.

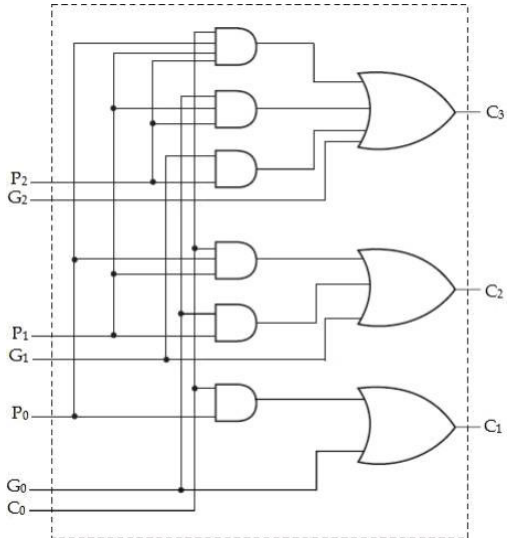
22. What is meant by carry propagation delay?

In Parallel adder, all the bits of the augend and the addend are available for computation at the same time. The carry output of each full-adder stage is connected to the carry input of the next high-order stage. Since each bit of the sum output depends on the value of the input carry, time delay occurs in the addition process. This time delay is called as carry propagation delay.

23. What is look ahead-carry addition?

The sum bit generated in the last position (MSB) depends on the carry that was generated by the addition in the previous position. i.e., the adder will not produce correct result until LSB carry has propagated through the intermediate full-adders. This represents a time delay that depends on the propagation delay produced in an each full-adder. The method of speeding up this process by eliminating inter stage carry delay is called look ahead-carry addition. This method utilizes logic gates to look at the lower order bits of the augend and addend to see if a higher-order carry is to be generated. It uses two functions: carry generate and carry propagate.

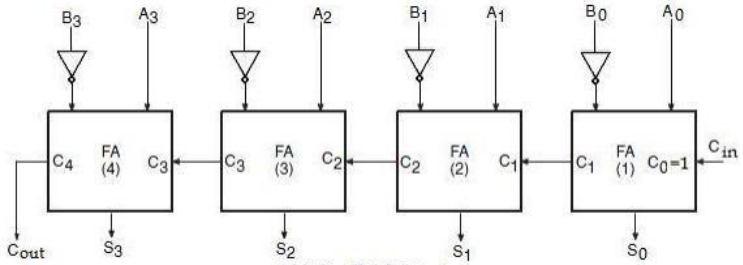
24. Draw the Logic diagram of Carry Look-ahead Generator.



25. What is Binary Subtractor (Parallel Subtractor)?

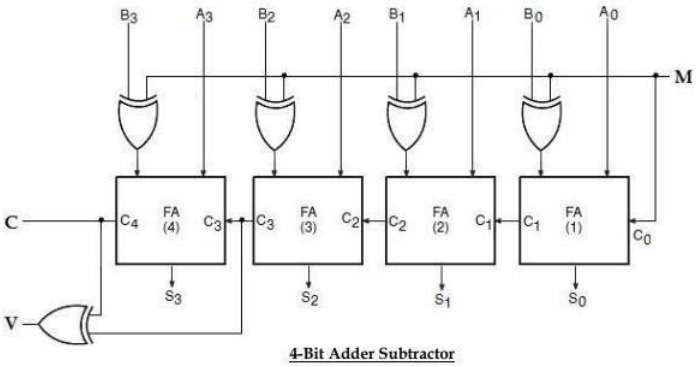
The subtraction of unsigned binary numbers can be done most conveniently by means of complements. The subtraction A-B can be done by taking the 2‘s complement of B and adding it to A. The 2‘s complement can be obtained by taking the 1‘s complement and adding 1 to the least significant pair of bits. The 1‘s complement can be implemented with inverters and a 1 can be added to the sum through the input carry.

26. Draw the logic diagram of 4-bit Parallel Subtractor.



4-bit Parallel Subtractor

27. Draw the logic diagram of 4-Bit Adder Subtractor.



4-Bit Adder Subtractor

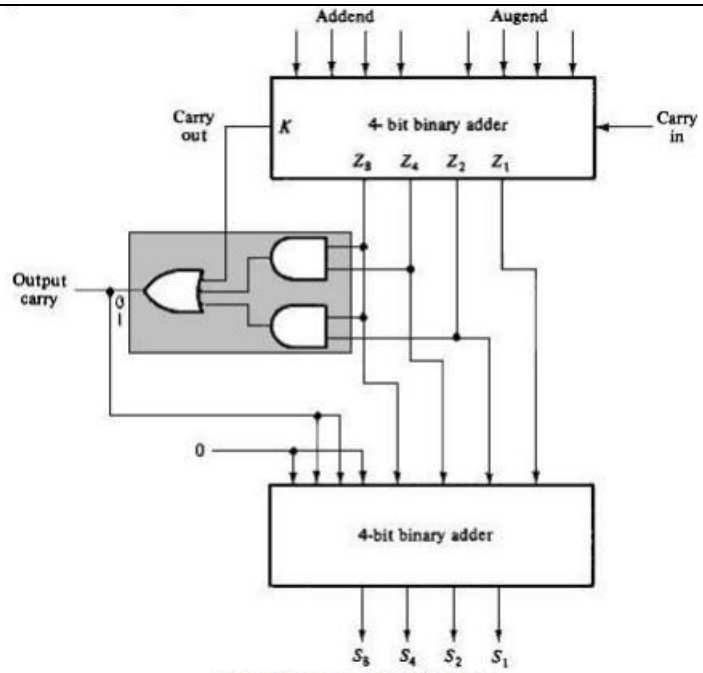
28. Define Decimal Adder (BCD Adder).

The digital system handles the decimal number in the form of binary coded decimal numbers (BCD). A BCD adder is a circuit that adds two BCD bits and produces a sum digit also in BCD.

29. Write the requirement to implement BCD adder.

- 4-bit binary adder for initial addition
- Logic circuit to detect sum greater than 9 and
- One more 4-bit adder to add 01102 in the sum if the sum is greater than 9 or carry is 1.

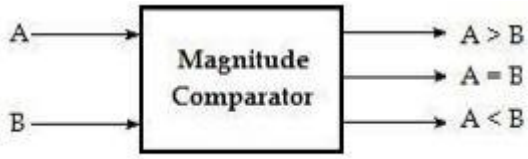
30. Draw the Block diagram of BCD adder.



Block diagram of BCD adder

31. Define magnitude comparator. (April/May 2015)

A magnitude comparator is a combinational circuit that compares two given numbers (A and B) and determines whether one is equal to, less than or greater than the other. The output is in the form of three binary variables representing the conditions A = B, A>B and A<B, if A and B are the two numbers being compared.



Block diagram of magnitude comparator

32. Draw the truth table of 2-bit comparator.

Truth table:

Inputs				Outputs		
A_3	A_2	A_1	A_0	A>B	A=B	A<B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

UNIT II SYNCHRONOUS SEQUENTIAL LOGIC

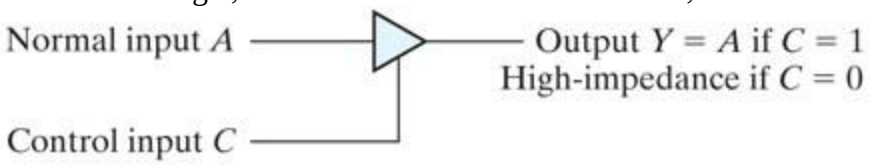
PART A

1. Define flip-flop.

The storage elements (memory) used in clocked sequential circuits are called *flip-flops.* A flip-flop is a binary storagedevice capable of storing one bit of information. In a stable state, the output of a flip-flop is either 0 or 1. A sequentialcircuit may use many flip-flops to store as many bits as necessary.

2. With graphic symbol define three-state gate & three-state buffer.

A multiplexer can be constructed with three-state gates—digital circuits that exhibit three states. Two of the states are signals equivalent to logic 1 and logic 0 as in a conventional gate. The third state is a *high impedance* state in which (1) the logic behaves like an open circuit, which means that the output appears tobe disconnected, (2) the circuit has no logic significance, and (3) the circuit connected to the output of the three-state gate is not affected by the inputs to the gate. Three-state gates may perform any conventional logic, such as AND or NAND. However, the one most commonly used is the buffer gate.



3. Define the different types of sequential circuit.

A *synchronous* sequential circuit is a system whose behavior can be defined from the knowledge of itssignals at discrete instants of time. The behavior of an *asynchronous* sequential circuit depends upon the

input signals at any instant of time *and* the order in which the inputs change. The storage elements commonlyused in asynchronous sequential circuits are time-delay devices. The storage capability of a time-delay device varies with the time it takes for the signal to propagate through the device.

J	*K*	Q(t+1)	
0	0	*Q*(*t*)	No change
0	1	0	Reset
1	0	1	Set
1	1	Q′(t)	Complement

D Flip-Flop			*T* Flip-Flop		
D	Q(t+1)		*T*	Q(t+1)	
0	0	Reset	0	*Q*(*t*)	No change
1	1	Set	1	Q′(t)	Complement

4. Draw the characteristic tables of D, T &JK Flip-flops. .

5. Describe the functionality of a D-type flip-flop.

A *D*-type flip-flop has a *D* (data) input, a clock input, and possibly asynchronous or synchronous clear (reset) or set signal. If *set* or *clear* are not asserted, the clock signal synchronizes the transfer of *D* to *Q*, the output. If *set* or *reset* are asynchronous, their action controls the flip-flop independently of the clock. *set* causes the output to be 1; *reset* causes the output to be 0. If *set* or *reset* are synchronous, their action has effect at the synchronizing edge of the clock.

6. Write the characteristic equations of D, T &JK Flip-flops.

   $Q(t+1)=D$

   $Q(t+1)=JQ'+K'Q$

   $Q(t+1)=T\oplus Q=TQ'+T'Q$

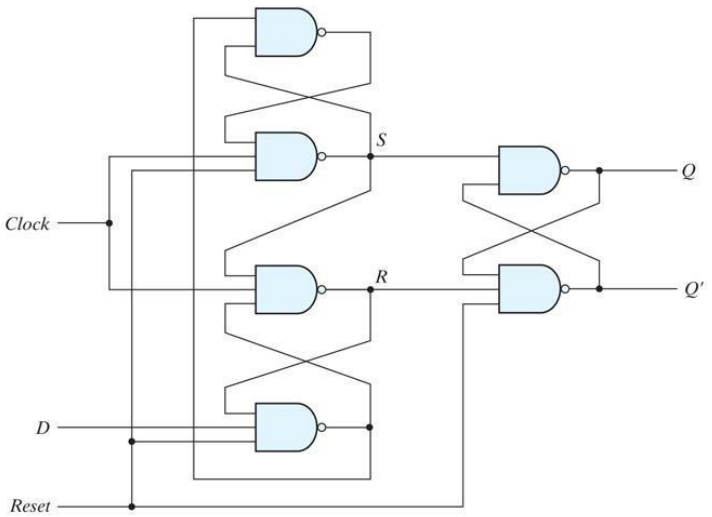
7. Differentiate direct set from direct reset.

   Some flip-flops have asynchronous inputs that are used to force the flipflop to a particular state independently of the clock. The input that sets the flip-flop to 1 is called *preset* or *direct set*. The input that clears the flipflop to 0 is called *clear* or *direct reset*. When power is turned on in a digital system, the state of the flip-flops is unknown. The direct inputs are useful for bringing all flip-flops in the system to a known startingstate prior to the clocked operation.

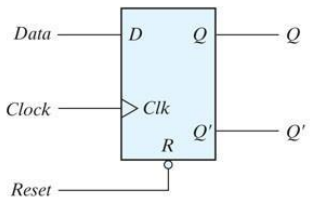
8. Write the use of direct inputs.

   When power is turned on in a digital system, the state of the flip-flops is unknown. The direct inputs are useful for bringing all flip-flops in the system to a known starting state prior to the clocked operation.

9. Draw the graphic symbol and write the function table of D flip-flop with asynchronous reset.



(a) Circuit diagram



(b) Graphic symbol

R	Clk	D	Q	Q'
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

(c) Function table

10. Simplify the Boolean function $F(x,y,z)=\Sigma(2,3,4,5)$

11. For the Boolean function

    $F=A'C+A'B+AB'C+BC$Express this function as a sum of minterms.

    Find the minimal sum-of-products expression.

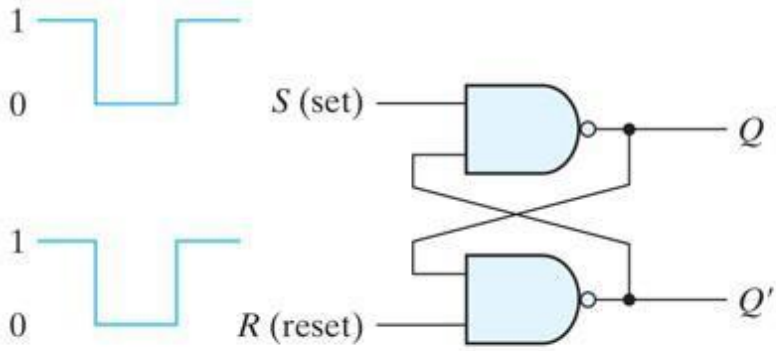
    12. For the Boolean function $F(x,y,z)=\mathbf{xy'z+x'y+x'z+yz}$

    a) Express this function as a sum of minterms.

    b) Find the minimal sum-of-products expression.

13. Simplify the Boolean function F(x,y,z)=Σ(3,4,6,7)

14. Define SR latch.



(a) Logic diagram

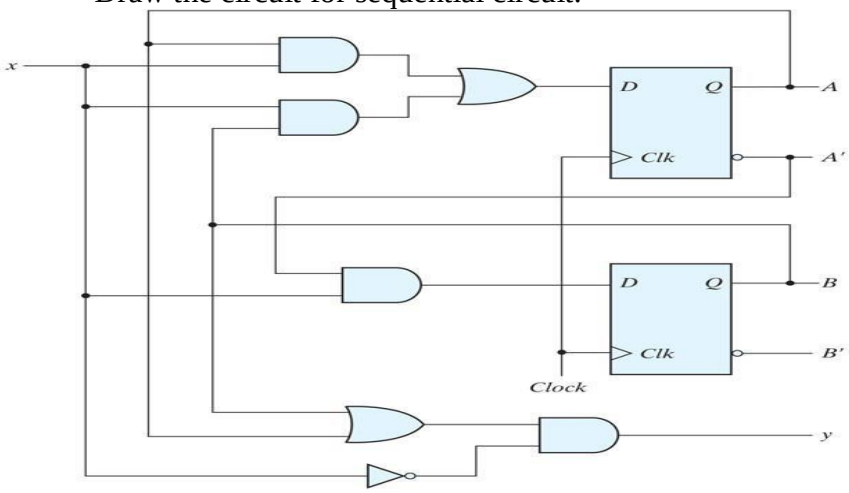
S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

(b) Function table

15. Define state equation.

The behavior of a clocked sequential circuit can be described algebraically by means of state equations. A *state equation* (also called a *transition equation*) specifies the next state as a function of the present state and inputs.

16. Draw the circuit for sequential circuit.



17. Define state table.

The time sequence of inputs, outputs, and flip-flop states can be enumerated in a *state table* (sometimes called a *transitiontable*).

18. Define state diagram.

The information available in a state table can be represented graphically in the form of a state diagram. In this type of diagram, a state is represented by a circle, and the (clock-triggered) transitions between states are indicated by directed lines connecting the circles. Each line originates at a present state and terminates at a next state, depending on the input applied when the circuit is in the present state.

The state diagram gives a pictorial view of state transitions and is the form more suitable for human interpretation of the circuit's operation.

19. Define *output* equations.

The part of the combinational circuit that generates external outputs is described algebraically by a set of Boolean functions called *output equations*.

20. Define input equations.

The part of the circuit that generates the inputs to flip-flops is described algebraically by a set of Boolean functions called flip-flop *input equations* (or, sometimes, *excitation equations*).

UNIT III COMPUTER FUNDAMENTALS

1. Write the equation for the dynamic power required per transistor?

Power α ½ * capacitor load*voltage2 * Frequency switched

2. Classify the instructions based on the observations they perform and give one example to each category

a) Arithmetic Instruction
b) Logical
c) Data Transfer
d) Jump

3. What are the components of a computer system?

Input, Output, Memory, Datapath and control

4. How to represent Instruction in a computer system? Instructions are represented in a computer using three formatsR Type for Register (or) R Format
I Type for Immediate (or) I FormatJ Type for Jump (or) J Format

5.Distinguish between auto increment and auto decrement addressing mode.

Auto Increment mode	Auto Decrement mode
In this mode the effective address of the operand is the content of a register specified in the instruction. After accessing the operand the content of this register are automatically incremented to point to the next item in a list	In this mode the content of a register specified in the instruction are first automatically decremented and then used effectively.
It can be written as (Ri)+	It can be written as - (Ri)

6. What are the addressing modes?

Referenced specified in an instruction is known as addressing modes
n instruction into an effective address from where the operand is actually

The addressing modes are the following:

1. Immediate addressing, where the operand is a constant within the instruction itself
2. Register addressing, where the operand is a register
3. Base or displacement addressing, where the operand is at the memory location whose address is the sum of a register and a constantin the instruction
4. PC-relative addressing, where the branch address is the sum of the PC and a constant in the instruction
5. Pseudo direct addressing, where the jump address is the 26 bits of the instruction concatenated with the upper bits of the PC

7. State the Need for indirect addressing mode. Give an example.

In mode, where the operand is at the memory whose address is the sum of register and constant in the instruction. Ex: lw sto, 32 (ss3)

8. What is an instruction register (IR)?

An IR is the part of a CPU's control unit that holds the instruction currently being executed or decoded

9. What is instruction set architecture (ISA)?

-ISA is a hardware/software interface. The contract b/w software & hardware ISA is the part of the computerarchitecture related to programming including the data types, registers and addressing modes.

Give the formula for CPU execution time for a program?

CPU execution time for program=CPU clock cycles for a program x clock cycle time.

10. List out the methods used to improve system performance.

The methods used to improve system performance are 1. Processor clock 2. Basic Performance Equation 3. Pipelining 4.Clock rate5.Instruction set 6.Compiler.

11. Define Opcode and Operand .

Opcode is the portion of a machine language instruction that specifies the operation to be performed.
Ex: MOV AX, 1000H ; here MOV specifies the movement of 1000H in the AX

register. Operand is a quality on which operation is performed. Here AX and 1000H are the operands.

12. Define MIPS Rate and Throughput Rate.

MIPS Rate: The rate at which the instructions are executed at a given time. Throughput: The total amount of work done in a giventime.

Throughput rate: The rate at which the total amount of work done at a given time.

13. State Amdhal's law.

Amdahl's law, also known as Amdahl's argument, is used to find the maximum expected improvement to an overall system when onlypart of the system is improved. It is often used in parallel computing to predict the theoretical maximum speedup using multiple processors.

14. Define CPI

The term Clock Cycles Per Instruction Which is the average number of clock cycles each instruction takes to execute, is often abbreviated as CPI.

CPI= CPU clock cycles/Instruction count.

15. State and explain the performance equation?

The average number of basic steps needed to execute one machine instruction is S, where each basic step is completed in one clock cycle. If the clock cycle rate is R cycles per second, the program execution time is given by T = (N x S) / R This is often referred to as the basic performance equation.

16. How CPU execution time for a program is calculated?

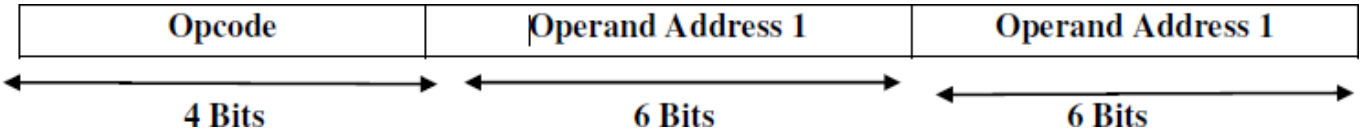
CPU execution time for a program= CPU clock cycles for a program*Clock Cycle Time

$$\text{CPU execution time for a program} = \frac{\text{CPU Clock cycles for a program}}{\text{Clock Rate}}$$

17. How to represent Instruction in a Computer System?

The Instruction has three fields:

1. Opcode
2. Operand Address 1
3. Operand Address 2



18. Define MAR.

MAR: Memory address register is used to hold the address of the location to or from which data are to be transferred.

19. Define Word length.

The number of bits in each word is often referred to as the word length of the computer. Word length ranges from 16 to 64 bits.

20. Write the rules to perform addition on floating point numbers.Add/Subtract Rule

1. Choose the number with the smaller exponent and shift its mantissa right a number of steps equal to the difference in exponents.
2. Set the exponent of the result equal to the larger exponent.
3. Perform addition/subtraction on the mantissas and determine the sign of the result.
4. Normalize the resulting value, if necessary.

22. What is arithmetic overflow?

Overflow occurs when there are insufficient bits in a binary number representation to portray the results of an arithmetic operation. Overflow occurs because computer arithmetic is not closed with respect to addition, subtraction, multiplication or division. Overflowcannot occur in addition, if the operands have different sign.

23. Subtract (11011)2 -(10011)2 using 2's complement.

2's complement of 1

1's complement

```
10000
01111
11010
01111
 1011
```

1000

2's complement

```
01111
10000
```

Subtraction

```
11010
10000
 1010
```

Ans: → 101000.

24. Subtract (11010)2 -(10000)2 usin ment method.

25. Divide (1001010)2 ÷

```
        1 1 1
.1000 | 1 0 0 1 0 1 0 |
        1 0 0 0 |
        1 0 1 0 1
          1 0 0 0
          1 1 0 1 0
            1 0 0 0
        1 1 1 0 0 1 0 – Remainder
```

UNIT 4 PROCESSOR

1. Define a data path in a CPU.
A unit used to operate on or hold data within a processor. In the MIPS implementation, the data path elements include the instructionand data memories, the register file, the ALU, and adders.

2. Name the control signals required to perform arithmetic operations.
(a) Reg Not (b) Reg write (c) ALL src (d) PCsrc (e) Mem Read (f) Mem to Reg

3. What is meant by pipeline bubble?
Pipeline bubble or pipeline stall is a delay in execution of an instruction in an instruction pipeline in order to resolve the hazard. During the decoding stage, the control unit will determine if the decoded instruction reads from a register that the instruction currently in the

4. What is the ideal CPI of a pipelined processor?
The ideal CPI on a pipelined processor is almost always 1. Hence, we can compute the pipelined CPI:
CPI pipelined = Ideal CPI + Pipeline stall clock cycles per instruction = 1 + Pipeline stall clock cycles per instruction

5. Mention the various types of pipelining.
It is divided into 2 categories:
- Arithmetic Pipeline
- Instruction Pipeline

6. Mention the various phase in executing an instruction.
- Fetch
- Decode
- Execute
- Memory
- Write Back

7. What are the advantages of pipelining?
The cycle time of the processor is reduced, thus increasing instruction issue rate in most cases. Some combinational circuits such as adders or multipliers can be made faster by adding more circuitry. If pipelining is used instead it can save circuitry and also a more complex combinational circuit.

8. What is meant by branch prediction?
Branch prediction, Predict the next fetch address. There are two branch prediction techniques:

- Static branch prediction
- Dynamic branch prediction

The performance of branch prediction technique depends on

- Accuracy
- Cost

9. Give the features of the addressing modes suitable for pipelining.

- Access to an operand does not require more than one access to the memory
- Only load and store instructions access memory operands
- The addressing modes used do not have side effects

10. What is loop unrolling?

A simple scheme for increasing the number of instructions relative to the branch and overhead instructions is loop unrolling. Unrollingsimply replicates the loop body multiples times, adjusting the loop termination code.

11. What is the role of cache memory in pipeline?

The use of cache memory is to solve the memory access problem. When cache is included in the processor the access time to the cacheis usually the same time needed to perform other basic operation inside the processor.

12. Name the methods for generating the control signals.

The methods for generating the control signals are:

1) Hardwired control
2) Microprogrammed control

13. What are the two main approaches to hardware multithreading?

There are two main approaches to hardware multithreading. Fine-grained multithreading switches between threads on each instruction, resulting in interleaved execution of multiple threads. This interleaving is often done in a round-robin fashion, skipping any threads that are stalled at that clock cycle. Coarse-grained multithreading is an alternative to fine-grained multithreading. It switches threads only on costly stalls, such as last-level cache misses.

14.Define data hazard.

A data hazard is any condition in which either the source or the destination operands of an instruction are not available at the time expected in the pipeline. A data hazard is a situation in which the pipeline is stalled because the data to be operated on are delayed for some reason.

15.List the two steps involved in executing an instruction.

- Fetch the Instruction
- Fetch the operands

16. Define data path element.

A unit used to operate on or hold data within a processor. In the MIPS Implementation, the data path elements include the instruction and data memories, theALU, register file, the and adders.

17.List the five stages of instruction execution.

1. IF: Instruction fetch
2. ID: Instruction decode and register file read
3. EX: Execution or address calculation
4. MEM: Data memory access
5. WB: Write back

18.How data hazard can be prevented in pipelining?

Data hazards in the instruction pipelining can prevented by the following techniques.

a) Operand Forwarding b) Software Approach

19.**Define Pipelining.**

In order to reduce the overall processing time several instructions are being executed simultaneously. This process is termed as pipelining.

20.**Define instruction hazard or control hazard.**

A pipeline may also be stalled because of the delayed in the availability of an instruction. This may be a result of a miss in the catch, requiring the instruction to be fetched from the main memory. Such hazard are often called control hazard..

21,**Give the major limitation of pipelining technique**.

If an instruction is stalled in the pipeline, no later instructions can proceed. Thus, if there is a dependency between two closely spaced instructions in the pipeline, it will stall.

UNIT 5 – MEMORY AND I/O

1. What are the various memory technologies?

- Main memory-SRAM semiconductor memory
- Main memory-DRAM semiconductor memory
- Flash semiconductor memory
- Magnetic disk

2. Define Hit Ratio.

Hit Ratio is the fraction memory access found in the upper level .It is often used as a measure of the performance of the memory hierarchy.

3. Distinguish SRAM and DRAM.

SRAMs are simply integrated circuits that are memory arrays with the single access port that can provide either read or a write. SRAMs have a fixed access time to any datum. SRAMs don't need to refresh and so the access time is very close to the cycle time. SRAMs typically use 6 to 8 transistors per bit to prevent the information from being disturbed when read. SRAM needs only minimal power to retain the charge in standby mode.

In a DRAM the value kept in a cell is stored as a charge in a capacitor. A single transistor is then used to access this stored charge, either to read the value or to over write the charge stored there. Because DRAMs use only a single transistor per bit of storage, they are much denser and cheaper per bit than SRAM. As DRAM store charge on a capacitor, it cannot be kept indefinitely and must periodically be refreshed.

4. What is virtual memory?

A technique that uses main memory as a "cache" for secondary storage. Two major motive for Virtual Memory

- To allow efficiency and safe sharing of memory among multiple programs
- To remove the programming burdens of a small, limited amount of main memory

5. Define memory hierarchy.

In computer architecture memory hierarchy is a concept used for storing and discussing performance issues in computer architectural design, algorithm predictions and the lower level programming constructs such as involving locality of reference. The memory hierarchy in computer storage distinguishes each level in the hierarchy by response time. Since response time, complexity and capacity are related, the levels may also be distinguished by their performance and controlling technologies.

6. State the advantages of virtual memory

- Easier memory management
- Provides memory isolation/ protection

7. What is cache memory?

Cache memory is random access memory (RAM) that a computer microprocessor can access more quickly that it can access regular RAMs

8. Define memory interleaving.

Memory Interleaving is a design made to compensate for the relatively slow speed of dynamic RAM by spreading memory address evenly across memory banks.

9. Summarize the sequence of the events involved in handling an interrupt request from a single device.

- The device raises an interrupt request
- The processor interrupts the program currently being executed
- Interrupts are disabled by changing the control bits in the PS
- The action requested by the interrupts is performed by ISR
- Interrupts are enabled and execution of the interrupted program is resumed

10. How many total bits are required for a direct map cache with 16KB of data and 4- word blocks, assuming a 32bit address?

Solution:

We know that 16KB is 4096, 4K words is 212 words.

Block size of 4 words (22), there are 1024 (210) blocks.

Each block has 4 x 32 = 128 bits of data plus a tag.

Thus the total catch size is: 210 x (128+(32-10-2-2)+1) = 210 x 147 = 147 bits

11. What is the use of DMA controller?

- Used for high speed I/O devices
- Device interface transfers data directly to or from the memory
- Processor not continuously involved

12. What is miss rate?

The **miss rate (1−hit rate) is the fraction of memory accesses** not found in the upper level.

13. State the advantages of the Virtual Memory.

1. Virtual memory makes application programming easier by hiding fragmentation of physical memory.
2. We can run more applications at once.

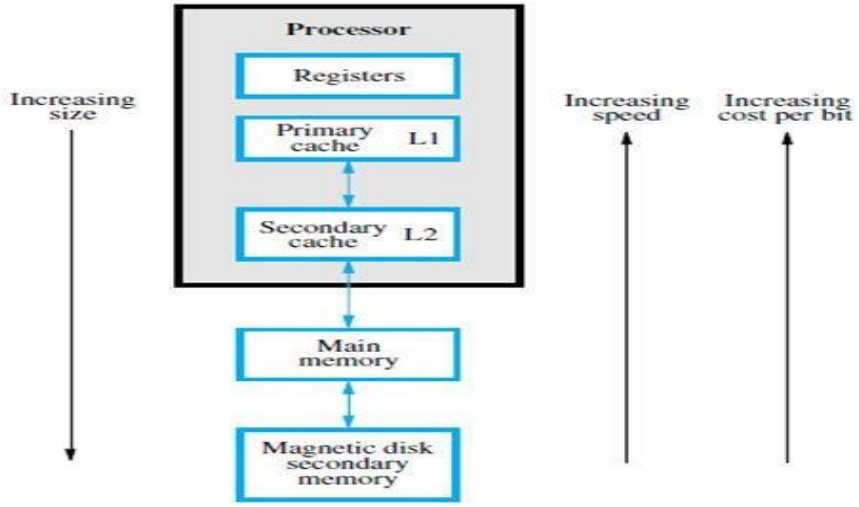
14. Differentiate Programmed I/O and Interrupt I/O

Sl. No	Programmed I/O	Interrupt I/O
1	During polling processor is busy and therefore have serious and decremental effect on system throughput.	Here the processor is allowed to execute its instruction in sequence and only stop to service I/O device when it is told to do so by the device itself. This increase system throughput.
2	It is implemented without interrupt hardware support	It is implemented using interrupt hardware support.
3	It does not depend on interrupt status.	Interrupt must be enabled to process
4	It does not need initialization of stack	It needs initialization of stack
5	System throughput decreases as number of I/O devices connected in the system increases	System throughput does not depend on number of I/O devices connected in the system.

15. What do you mean by memory mapped I/O?

In Memory mapped I/O, there is no specific input or output instructions. The CPU can manipulate I/O data residing in interface registers with the same instructions that are used to manipulate memory words.

16. Draw Memory Hierarchy in a typical computer system.



Memory hierarchy

17. What is meant by address mapping?

Address mapping is defined as the smallest unit of addressed data that can be mapped independently in an area of the virtual address space.

18. Protein string matching code has four days execution time on current machine doing integer instructions in 20 % of time,doing I/O in 35% of time and other operations in the remaining time. Which is the better trade off among the following two proposals? First: Compiler optimization that reduces number of integer instructions by 25% (assume each integer instructiontakes the same amount of time); Second: Hardware optimization that reduces the latency of each I/O operations from 6µs to 5 µs.

Solution:

4 days execution time on current machine 20% of time doing integer instructions 35% of time doing I/O

Speed up integer ops

X=0.2

S= (1/1-0.25)=1.33

Sint =1/(0.2/1.133+0.8) = 1.052

Speed up IO

X=0.35

S=6µs/5 µs = 1.2

Speeding Up IO is better

. **Define a super scalar processor**.

Super scalar processors attempts to issue multiple instruction per cycle. However, essential dependencies are specified by sequential ordering so operations must be processed in sequential order.

14. Give the key characteristics of GPUs from CPUs:

- GPUs are accelerators that supplement a CPU, so they do not need be able to perform all the tasks of a CPU. This role allows them to dedicate all their resources to graphics. It's fine for GPUs to perform some tasks poorly or not at all, given that in a system with both a CPU and a GPU, the CPU can do them if needed.
- The GPU problems sizes are typically hundreds of megabytes to gigabytes, but not hundreds of gigabytes to terabytes.

15. What is Cluster?

Clusters are generally collections of computer connected to each other over their I/O interconnect via standard network switches and cables. Clusters are the best example of message passing parallel computer.

16. What are the three major distinctions Warehouse Scale computers have?

- Ample, easy parallelism
- Operational Costs Count
- Scale and the Opportunities/Problems Associated with Scale

17. Compare UMA and NUMA Multiprocessors.

Uniform Memory Access(UMA) Multiprocessors: A Multiprocessors in which latency to any word in main memory is about the same no matter which processor requests the access.

Non- Uniform Memory Access(NUMA) Multiprocessors: A type of single address space multiprocessor in which some memory accesses are much faster than others depending on which processor asks for which word.

18. What are the classifications made by Flynn's?

The classifications defined by Flynn are based upon the number of concurrent instruction (or control) and data streams available in the architecture.

1. Single Instruction, Single Data stream (SISD)
2. Single Instruction ,Multiple data stream(SIMD)
3. Multiple Instruction, Single Data stream (MISD)
4. Multiple Instruction, Multiple Data streams (MIMD)
5. single program, multiple data (SPMD)

19. what is the basic philosophy of Vector Architecture?

The basic philosophy of vector architecture is to collect data elements from memory, put them in order into a large set of registers, operate on them sequentially in registers using **pipelined execution units**, and then write the results back to memory.

20. Define task level parallelism.

High performance can mean high throughput for independent task. Utilizing multiple processors by running independent programs simultaneously is called as **task level parallelism** or process level parallelism.

Part (B&C)UNIT 1

1. Explain in detail about half-adder. Design a full adder using 2 half adders.

Ans:1.Truth Table
2.K Map
3.Logic Diagram

2 a)Construct a 4 to 16 line decoder with an enable input using five 2 to 4 line decoders with enable inputs
b)Design a 4 bit adder / subtractor circuit and explain.

Ans:1.Truth Table
2.Logic diagram
3.Applications

3.What is BCD adder? Design an adder to perform arithmetic addition of two decimaldigits in BCD

1.Truth Table
2.Example

4 . Design an one-bit and 2-bit magnitude comparator.

1.Truth Table
2.Logic diagram

FOUR - VARIABLE K-MAP.

* In a four-variable k-map, there are 16 squares (2^4), one for each minterms.

* In a 4-variable k-map,

→ One square represents one minterm, giving a term with four literals.

→ Two adjacent squares represent a term with three literals.

→ Four adjacent squares represent a term with two literals.

→ Eight adjacent squares represent a term with one literal.

→ Sixteen adjacent squares represent a function that is always 1.

* Four variable k-map Representation *

m0	m1	m3	m2
m4	m5	m7	m6
m_{12}	m_{13}	m_{15}	m14
m8	m9	m11	m10

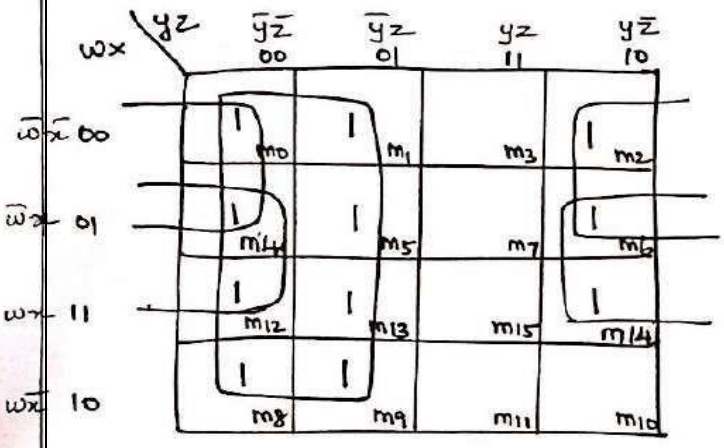
AB \ CD	$\bar{C}\bar{D}$ 00	$\bar{C}D$ 01	CD 11	$C\bar{D}$ 10
$\bar{A}\bar{B}$ 00	m0 $\bar{A}\bar{B}\bar{C}\bar{D}$	m1 $\bar{A}\bar{B}\bar{C}D$	m3 $\bar{A}\bar{B}CD$	m2 $\bar{A}\bar{B}C\bar{D}$
$\bar{A}B$ 01	m4 $\bar{A}B\bar{C}\bar{D}$	m5 $\bar{A}B\bar{C}D$	m7 $\bar{A}BCD$	m6 $\bar{A}BC\bar{D}$
AB 11	m12 $AB\bar{C}\bar{D}$	m13 $AB\bar{C}D$	m15 $ABCD$	m14 $ABC\bar{D}$
$A\bar{B}$ 10	m8 $A\bar{B}\bar{C}\bar{D}$	m9 $A\bar{B}\bar{C}D$	m11 $A\bar{B}CD$	m10 $A\bar{B}C\bar{D}$

① Simplify the Boolean function

$F(w,x,y,z) = \Sigma\{0,1,2,4,5,6,8,9,12,13,14\}$ using k-map.

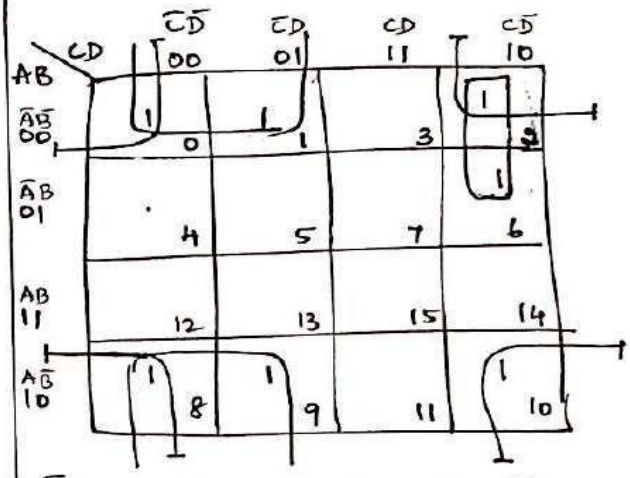
Solution :

Since the function has four variables, a four-variable map must be used. The minterms listed in the sum are marked by 1's in the map.



[Group 1 (0,1,4,5,12,13, 8,9) = $\bar{y}$
Group 2 (0,4, 2,6) = $\bar{w}\bar{z}$
Group 3 (4,12,6,14) = $x\bar{z}$]

∴ Simplified function $F = \bar{y} + \bar{w}\bar{z} + x\bar{z}$



[Group1 (0,1,8,9) = $\bar{B}\bar{C}$
Group 2 (2,6) = $\bar{A}C\bar{D}$
Group 3 (0,2,8,10) = $\bar{B}\bar{D}$]

∴ Simplified function $F = \bar{B}\bar{C} + \bar{A}C\bar{D} + \bar{B}\bar{D}$

② Simplify the function $F = \bar{A}\bar{B}\bar{C} + \bar{B}C\bar{D} + \bar{A}BC\bar{D} + A\bar{B}\bar{C}$ using k-map.

Solution:

* The given function F should be expressed as sum of minterms.

$$F = \bar{A}\bar{B}\bar{C} + \bar{B}C\bar{D} + \bar{A}BC\bar{D} + A\bar{B}\bar{C}$$

$$= \bar{A}\bar{B}\bar{C}(D+\bar{D}) + \bar{B}C\bar{D}(A+\bar{A}) + \bar{A}BC\bar{D} + A\bar{B}\bar{C}(D+\bar{D})$$

$$= \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}BC\bar{D} + A\bar{B}\bar{C}D + A\bar{B}\bar{C}\bar{D}$$

$$= 0001 + 0000 + 1010 + 0010 + 0110 + 1001 + 1000$$

$$= m_1 + m_0 + m_{10} + m_2 + m_6 + m_9 + m_8$$

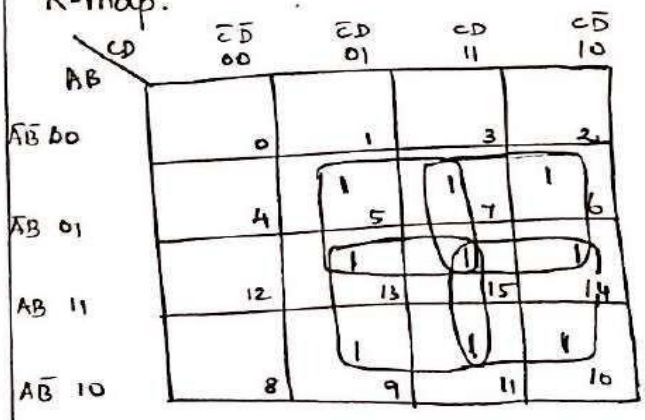
$$F = \Sigma(0,1,2,6,8,9,10)$$

* The minterms listed in the sum are marked by 1's in the map.

③ Find minimal SOP for $F = \Sigma m(5,6,7,9,10,11,13,14,15)$ using K-Map.

Solution:

Given the sum of minterms. These minterms are expressed as 1's in the K-map.



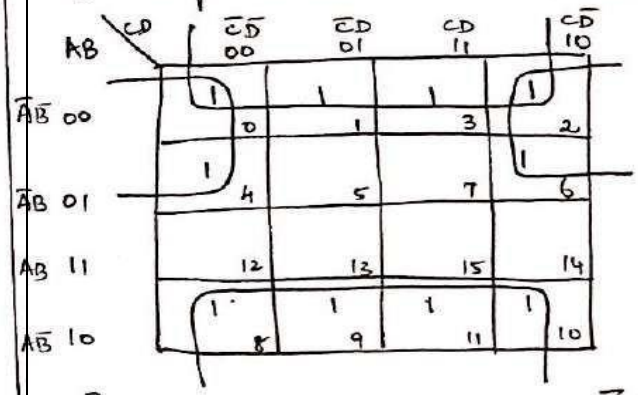
[$G_1(5,7,13,15) = BD$, $G_2(6,7,14,15) = BC$
$G_3(8,9,12,13) = AD$, $G_4(10,11,14,15) = AC$]

∴ Simplified $F = BD + BC + AC + AD$

④ Find minimal SOP for $F = \Sigma m(0,1,2,3,4,6,8,9,10,11)$ using K-map.

Solution:

Given the sum of minterms. These are represented as 1 in the map.



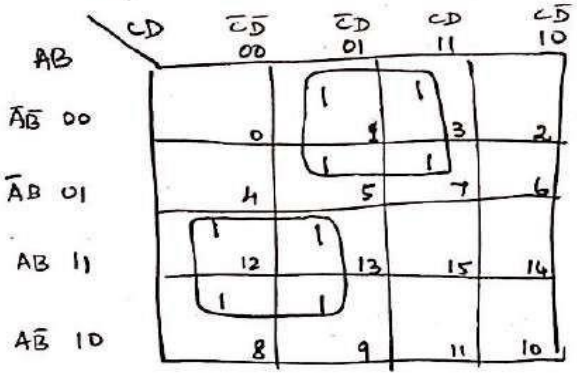
[Group1 $(0,1,2,3,8,9,10,11) = \bar{B}$
Group2 $(0,2,4,6) = \bar{A}\bar{D}$]

$\therefore$ Minimal SOP $= \bar{B} + \bar{A}\bar{D}$.

⑤ $Y = m_1 + m_3 + m_5 + m_7 + m_8 + m_9 + m_{12} + m_{13}$. Simplify the expression.

Solution:

Given the minterms $\Sigma(1,3,5,7,8,9,12,13)$. These minterms are represented as 1 in the map.



[Group1 $(1,3,5,7) = \bar{A}D$, Group2 $(8,9,12,13) = A\bar{C}$]

$\therefore$ Simplified function $F = \bar{A}D + A\bar{C}$

⑥ Plot the logical expression $ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C + AB$ on a 4-variable map and reduce it.

Solution:

Find the sum of minterms for the logical expression

$Y = ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C + AB$

$= ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C(D+\bar{D}) + AB(C+\bar{C})(D+\bar{D})$

$= ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + ABC(D+\bar{D}) + AB\bar{C}(D+\bar{D})$

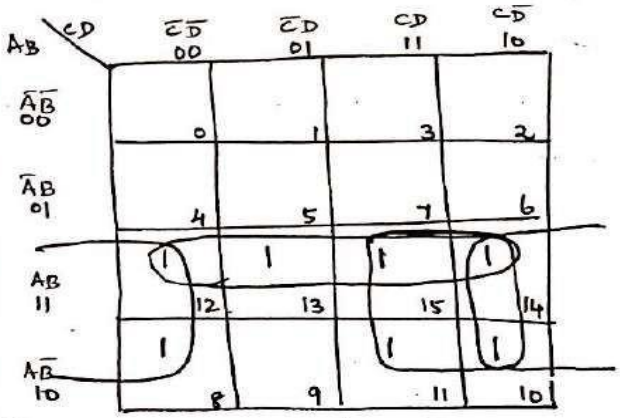
$= ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + ABCD + ABC\bar{D} + AB\bar{C}D + AB\bar{C}\bar{D}$.

$= ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + ABC\bar{D} + AB\bar{C}D + AB\bar{C}\bar{D}$

$= 1111 + 1000 + 1011 + 1010 + 1110 + 1101 + 1100$

$= m_{15} + m_8 + m_{11} + m_{10} + m_{14} + m_{13} + m_{12}$

Sum of minterms $= \Sigma(8,10,11,12,13,14,15)$



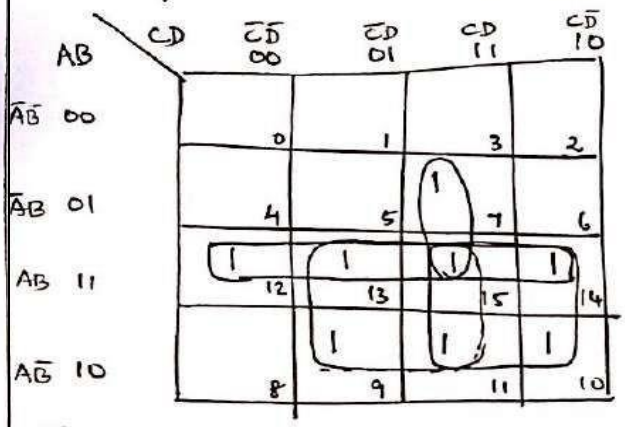
[Group1 $(12,13,14,15) = AB$
Group2 $(10,11,14,15) = AC$
Group3 $(8,10,12,14) = A\bar{D}$]

$\therefore$ The reduced expression $= AB + AC + A\bar{D}$

⑦ Simplify $y = \Sigma_m(7, 9, 10, 11, 12, 13, 14, 15)$ using k-map.

Solution:

Given the minterms. These minterms are represented as 1 on the k-map.



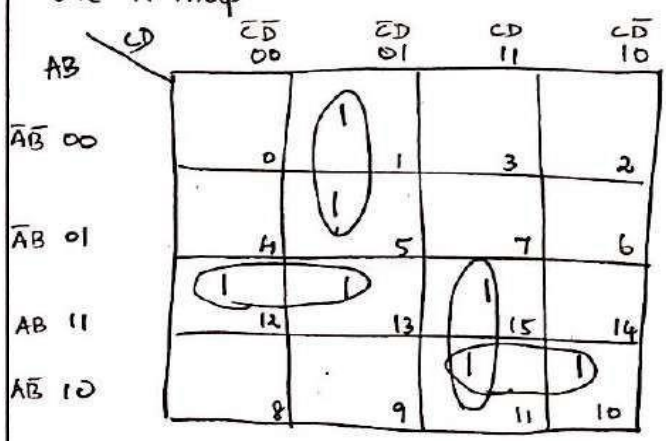
Group1 (12,13,14,15) = AB
Group2 (9,11,13,15) = AD
Group3 (10,11,14,15) = AC
Group4 (7,15) = BCD

∴ Simplified function F = $AB + AD + AC + BCD$

⑧ Simplify $y = m_1 + m_5 + m_{10} + m_{11} + m_{12} + m_{13} + m_{15}$.

Solution

Given the minterms $\Sigma(1, 5, 10, 11, 12, 13, 15)$ These minterms are represented as 1 on the k-map.



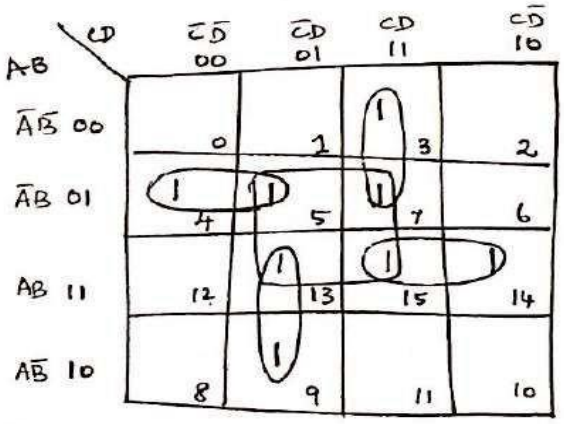
Group1 (1,5) = $\bar{A}\bar{C}D$
Group2 (12,13) = $AB\bar{C}$
Group3 (11,15) = ACD
Group4 (10,11) = $A\bar{B}C$

∴ Simplified $F = \bar{A}\bar{C}D + AB\bar{C} + ACD + A\bar{B}C$.

⑨ Simplify $y = \Sigma_m(3, 4, 5, 7, 9, 13, 14, 15)$ using k-map.

Solution:

Given the minterms $\Sigma(3, 4, 5, 7, 9, 13, 14, 15)$. These minterms are represented as 1 on the k-map.



G1 (5,7,13,15) = BD
G2 (3,7) = $\bar{A}CD$
G3 (4,5) = $\bar{A}B\bar{C}$
G4 (9,13) = $A\bar{C}D$
G5 (14,15) = ABC

* Here Group 1 is redundant because all the minterms in this group belongs to another group. The redundant group has to be ignored

$$y = \bar{A}CD + \bar{A}B\bar{C} + A\bar{C}D + ABC$$

(10) Minimize

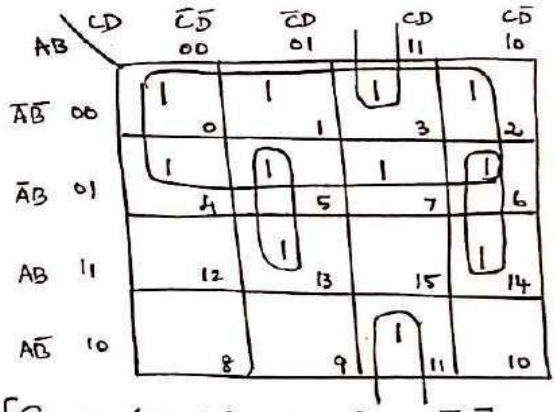
$$Y(A,B,C,D) = \Sigma(0,1,2,3,4,5,6,7,11,13,14)$$

using k-map.

Solution

Given minterms $\Sigma(0,1,2,3,4,5,6,7,11,13,14)$.

These minterms are represented as 1's in the k-map.



Group1 $(0,1,2,3,4,5,6,7) = \bar{A}$
Group2 $(3,11) = \bar{B}CD$
Group3 $(5,13) = B\bar{C}D$
Group4 $(6,14) = BC\bar{D}$

∴ Simplified function $Y =$

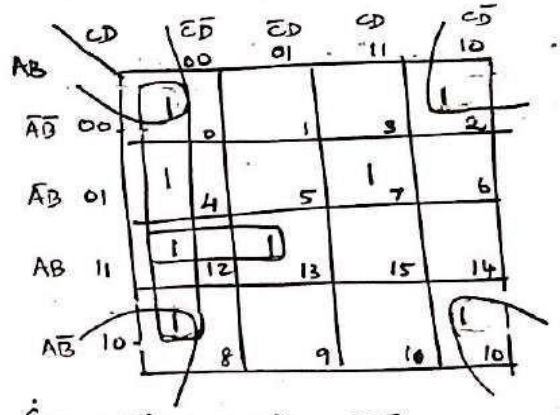
$$\bar{A} + B\bar{C}D + BC\bar{D} + \bar{B}CD$$

(11) Minimize $Y(A,B,C,D) = \Sigma(0,2,4,7,8,10,12,13)$ using k-map.

Solution:

Given minterms $\Sigma(0,2,4,7,8,10,12,13)$.

These minterms are represented as 1's in the k-map.



Group1 $(0,4,8,12) = \bar{C}\bar{D}$
Group2 $(12,13) = AB\bar{C}$
Group3 $(0,2,8,10) = \bar{B}\bar{D}$

∴ The Minimized function

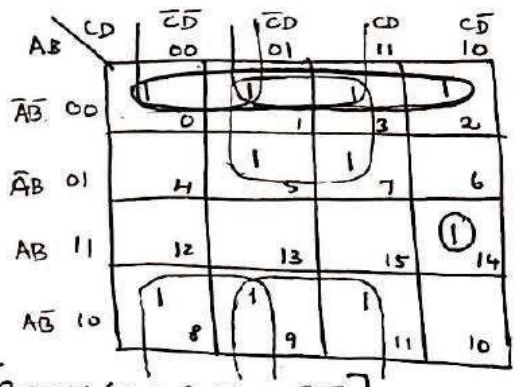
$$Y = \bar{C}\bar{D} + AB\bar{C} + \bar{B}\bar{D}.$$

(12) Minimize the following logic function using k-map.

$$Y = \Sigma\{0,1,2,3,5,7,8,9,11,14\}$$

Solution:

Given minterms. These minterms are represented as 1's in the k-map.



Group1 $(0,1,2,3) = \bar{A}\bar{B}$
Group2 $(1,3,5,7) = \bar{A}D$
Group3 $(0,1,8,9) = \bar{B}\bar{C}$
Group4 $(1,3,9,11) = \bar{B}D$
Group5 $(14) = ABC\bar{D}$

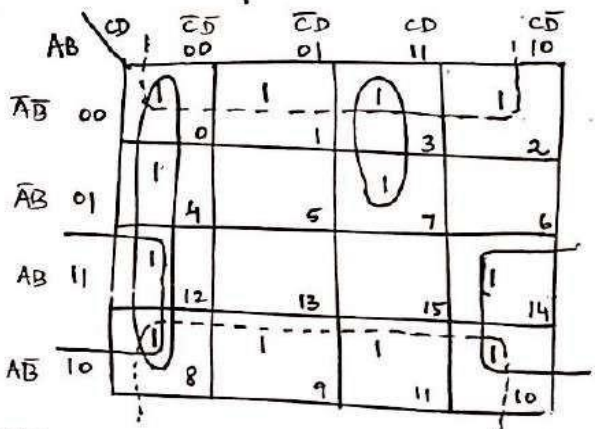
∴ Minimized function $Y =$

$$\bar{A}\bar{B} + \bar{A}D + \bar{B}\bar{C} + \bar{B}D + ABC\bar{D}.$$

(13) Simplify the Boolean expression $Y = \Sigma(0,1,2,3,4,7,8,9,10,11,12,14)$ using k-map.

Solution:

Given the minterms of the function. These minterms are represented as 1's on the k-map.



Group 1 $(0,1,2,3,8,9,10,11) = \bar{B}$
Group 2 $(0,4,8,12) = \bar{C}\bar{D}$
Group 3 $(8,10,12,14) = A\bar{D}$.
Group 4 $(3,7) = \bar{A}CD$

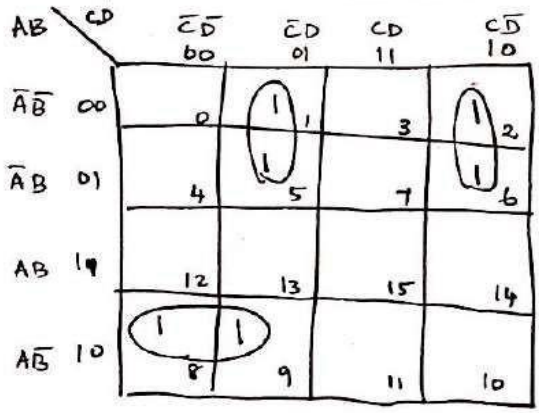
∴ Simplified function $Y =$

$\bar{B} + \bar{C}\bar{D} + A\bar{D} + \bar{A}CD$

(14) Simplify the Boolean expression $Y = \Sigma(1,2,5,6,8,9)$ using k-map.

Solution:

Given the minterms. These minterms are represented as 1's on the k-map.



Group1 $(1,5) = \bar{A}\bar{C}D$
Group2 $(2,6) = \bar{A}C\bar{D}$
Group3 $(8,9) = A\bar{B}\bar{C}$

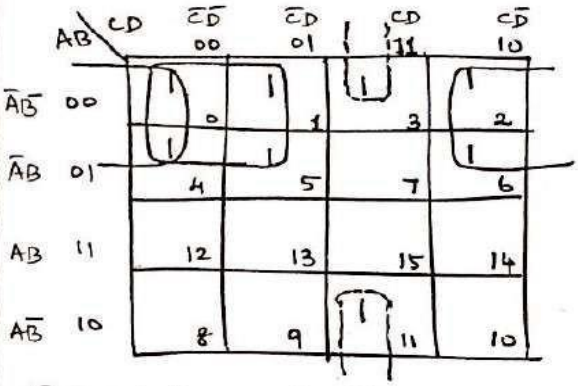
∴ The simplified Boolean expression

$Y = \bar{A}\bar{C}D + A\bar{B}\bar{C} + \bar{A}C\bar{D}$

(15) Simplify the Boolean expression $Y = \Sigma(0,1,2,3,4,5,6,11)$ using k-map.

Solution:

Given the minterms. These minterms are represented as 1's on the k-map



Group1 $(0,1,4,5) = \bar{A}\bar{C}$
Group 2 $(0,4,2,6) = \bar{A}\bar{D}$
Group 3 $(3,11) = \bar{B}CD$

∴ Simplified Boolean expression

$Y = \bar{A}\bar{C} + \bar{A}\bar{D} + \bar{B}CD$.

(15) Minimize the following Boolean expression using k-map $Y(A,B,C,D) = AB\bar{C} + BCD + BC\bar{D}$

Solution:

The given Boolean expression should be written as sum of minterms

$Y = AB\bar{C} + BCD + BC\bar{D}$

$= AB\bar{C}(D+\bar{D}) + BCD(A+\bar{A}) + BC\bar{D}(A+\bar{A})$

$= AB\bar{C}D + AB\bar{C}\bar{D} + ABCD + \bar{A}BCD + ABC\bar{D} + \bar{A}BC\bar{D}$

$= 1101 + 1100 + 1111 + 0111 + 1110 + 0110$

$= m_{13} + m_{12} + m_{15} + m_7 + m_{14} + m_6$

$\therefore Y = \Sigma(6, 7, 12, 13, 14, 15)$ (*)

Now, these minterms are represented as 1's on the k-map.

AB \ CD	$\bar{C}\bar{D}$ 00	$\bar{C}D$ 01	CD 11	$C\bar{D}$ 10
$\bar{A}\bar{B}$ 00	0	1	3	2
$\bar{A}B$ 01	4	5	1 7	1 6
AB 11	1 12	1 13	1 15	1 14
$A\bar{B}$ 10	8	9	11	10

[Group1 (6,7,14,15) = BC
Group2 (12,13,14,15) = AB]

$\therefore$ Simplified Function $Y = AB + BC$

(16) Minimize the following Boolean expression using k-map

$$Y = AB\bar{C} + BCD + B\bar{D}$$

Solution:

The given Boolean expression should be written as the sum of minterms.

$Y = AB\bar{C} + BCD + B\bar{D}$

$= AB\bar{C}D + AB\bar{C}\bar{D} + ABCD + \bar{A}BCD + AB\bar{D} + \bar{A}B\bar{D}(C+\bar{C})$

$= AB\bar{C}D + AB\bar{C}\bar{D} + ABCD + \bar{A}BCD + ABC\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}BC\bar{D} + \bar{A}\cancel{B}\bar{C}\bar{D}$

$= 1101 + 1100 + 1111 + 0111 + 1110 + 0110 + 0100$

$= m_{13} + m_{12} + m_{15} + m_7 + m_{14} + m_6 + m_4$

$\therefore Y = \Sigma(4, 6, 7, 12, 13, 14, 15)$ (*)

Now, these minterms are represented as 1's on the k-map.

AB \ CD	$\bar{C}\bar{D}$ 00	$\bar{C}D$ 01	CD 11	$C\bar{D}$ 10
$\bar{A}\bar{B}$ 00	0	1	2	3
$\bar{A}B$ 01	1 4	5	1 7	1 6
AB 11	1 12	1 13	1 15	1 14
$A\bar{B}$ 10	8	9	11	10

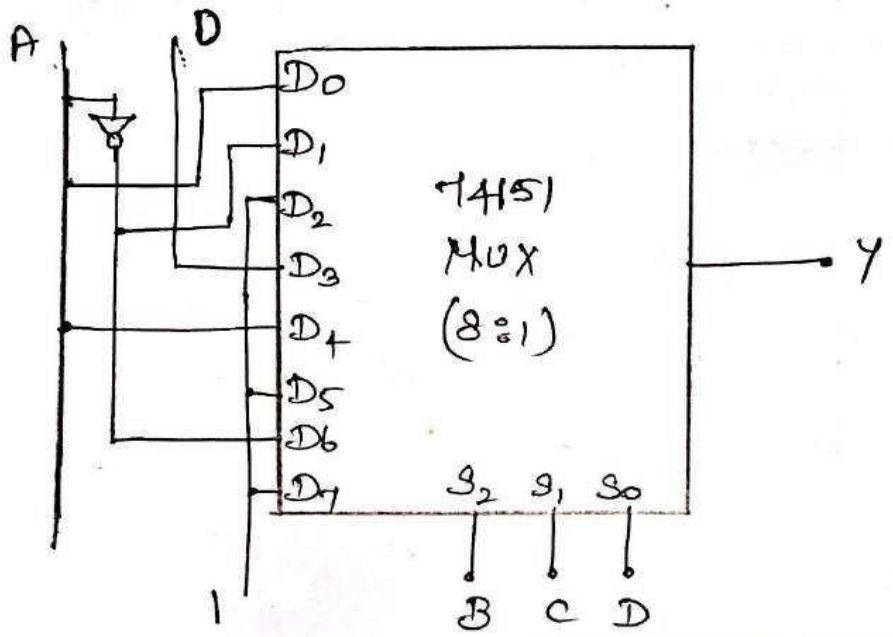
[Group1 (12,13,14,15) = AB
Group2 (4,6,12,14) = $B\bar{D}$
Group3 (6,7,14,15) = BC]

$\therefore$ Simplified Function Y

$= AB + B\bar{D} + BC$

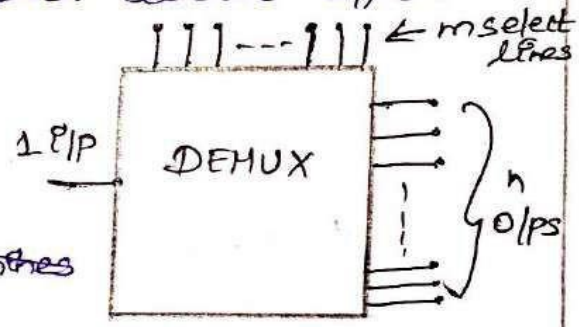
2.) Implement the following Boolean function using 4:1 Mux, $F(A,B,C) = \Sigma m(1,3,5,6)$

Multiplexer Implementation :-



DEMULTIPLEXERS :- [Data Distributors].

* One to Many
* It takes information from one i/p and transmit the same over several o/ps.
* 1 input
* – o/p lines
* m – Select lines
* ~~Select line determines to which o/p data~~



* At a time, only one o/p line is selected by the select lines and the i/p is transmitted to the selected o/p line.
* It is equivalent to single pole multiple way switch
* The enable i/p will enable the demux.
* Relation btw n o/p lines & m select lines is $\boxed{n = 2^m}$

UNIT 2

1.Explain the operation of JK FF, SR FF, T-FF and D-FF with a neat diagram. Also discuss their characteristic equation and excitation table.

Ans: Write the truth table ,characteristics table and Excitation table for each FlipFLop

2. Convert SR FlipFlop into JK FlipFlop

Ans:

a)Write the characteristics table for desired FF

b)From the characteristics table draw the Excitation table for Q(t),Q(t+1)

c)Write the simplified equation using k-map of the present state and desired FF i/p's .Draw the Equivalent diagram

3.Design a logic circuit diagram for 3-bit synchronous up-down counter.

4.Design a MOD-10 Synchronous counter using JK flip-flops. Write execution table and state table.

DESIGN OF ASYNCHRONOUS COUNTER :-

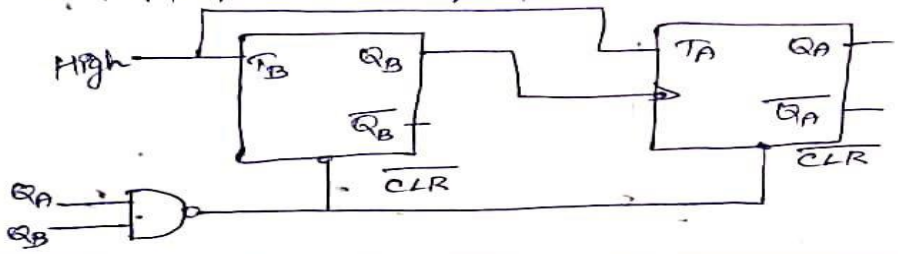
3 1.) Design MOD-3 ripple Counter :

Soln :- Sequence 0-1-2-0

Q_A	Q_B
0	0
0	1
1	0
1	1

$2 \le 2^2$

2 flipflops are required.



2.) Design MOD-10 /BCD/ Decade ripple Counter :-

i). using T FF ii) D-FF iii) JK-FF.

Soln :- Mod - N $N \le 2^n$

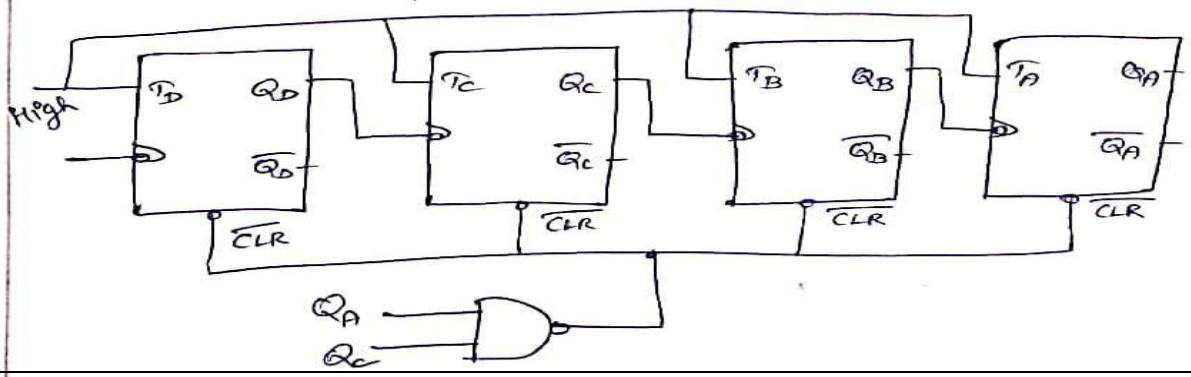
$10 \le 2^4$ $\Rightarrow$ 4 F.F's are required.

Using T Flipflop:

Sequence 0-1-2-3-4-5-6-7-8-9-0

when 10 Comes, It has to be reset to 0.

	Q_A	Q_B	Q_C	Q_D
10 $\rightarrow$	1	0	1	0



2.) Minimize the State table shown below:-

PS	NS, z (o/p)	
	$x=0$	$x=1$
A	B, 0	C, 0
B	B, 0	D, 0
C	B, 0	C, 0
D	E, 1	C, 0
E	B, 0	D, 0

Soln:-

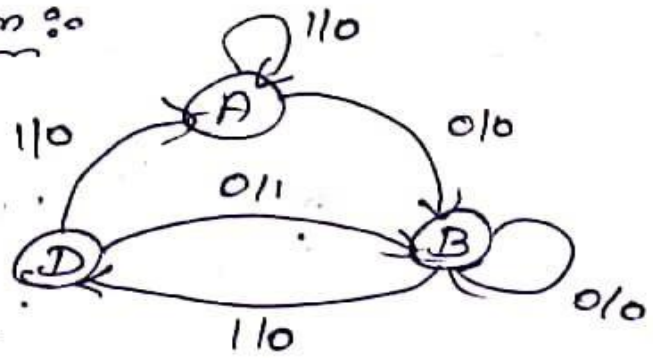
	PS	NS		O/P, z	
		$x=0$	$x=1$	$x=0$	$x=1$
→	A	B	~~C~~ A	0	0
→	B	B	D	0	0
X →	C	B	C	0	0
	D	~~E~~ B	~~C~~ A	1	0
X →	E	B	D	0	0

i.) $E \equiv B$ 2.) $C \equiv A$

Reduced state table:-

PS	N.S		O/P, z	
	$x=0$	$x=1$	$x=0$	$x=1$
A	B	A	0	0
B	B	D	0	0
D	B	A	1	0

State diagram:-



3.) Reduce the no. of states in the following State Table and (a) tabulate the reduced State table. (b) Starting from State a and I/P Sequence 0111001 0011 determine the O/P sequence for the given reduced table.

Soln:-

PS	NS		O/P	
	$x=0$	$x=1$	$x=0$	$x=1$
a	f	b	0	0
b	d	~~c~~ a	0	0
c	f	~~e~~ b	0	0
d	g	a	1	0
e	d	c	0	0
f	f	b	1	1
g	g	~~h~~ d	0	1
h	g	a	1	0

① $d \equiv h$
② $b \equiv e$
③ $a \equiv c$

Reduced State table / Transition table :-

PS	NS		O/P, Z	
	$x=0$	$x=1$	$x=0$	$x=1$
a	f	b	0	0
b	d	a	0	0
d	g	a	1	0
f	f	b	1	1
g	g	d	0	1

ii)

I/p sequence	State transition from a	O/p sequence.
0	a → f	0
1	f → b	1
1	b → a	0
1	a → b	0
0	b → d	0
0	d → g	1
1	g → d	1
0	d → g	1
0	g → g	0
1	g → d	1
1	d → a	0

STATE ASSIGNMENT :-

Three possible binary state assignments, [for three bits, possible states → 8]:

State	Assignment 1 Binary	Ass-2 Gray code	Ass-3 One-Hot code.
a	000	000	00001
b	001	001	00010
c	010	011	00100
d	011	010	01000
e	100	010	10000

* Only for five states.

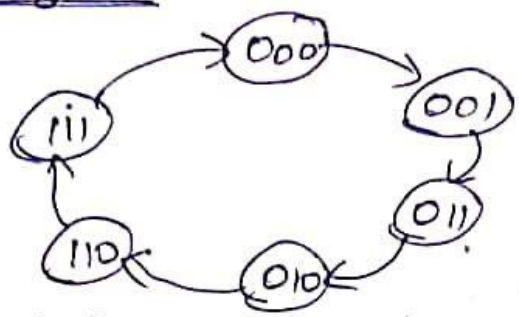
4.) Design of MOD-6 UNIT DISTANCE CODE / GRAY CODE Seq. ckt using JK Flipflop.

Soln: Unit distance code → Gray code.

$\therefore N = 6$; $6 \le 2^3$ 3 F.F's are required

Let p.s be (A, B, C).

State Diagram:



State Table:

PS			NS			F.F Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	1	0	X	1	X	X	0
0	1	0	0	1	0	0	X	X	0	X	1
0	1	1	1	1	0	1	X	X	0	0	X
1	0	0	1	1	1	X	0	X	0	1	X
1	0	1	0	0	0	X	1	X	1	X	1
1	1	0	X	X	X	X	X	X	X	X	X
1	1	1	X	X	X	X	X	X	X	X	X

Kmap for J_A:

A \ BC	00	01	11	10
0				1
1	X	X	X	X

$\boxed{J_A = B\bar{C}}$

K_A:

A \ BC	00	01	11	10
0	X	X	X	X
1	X	X	1	

$\boxed{K_A = C}$

J_B:

A \ BC	00	01	11	10
0		1	X	X
1	X	X	X	X

$\boxed{J_B = C}$

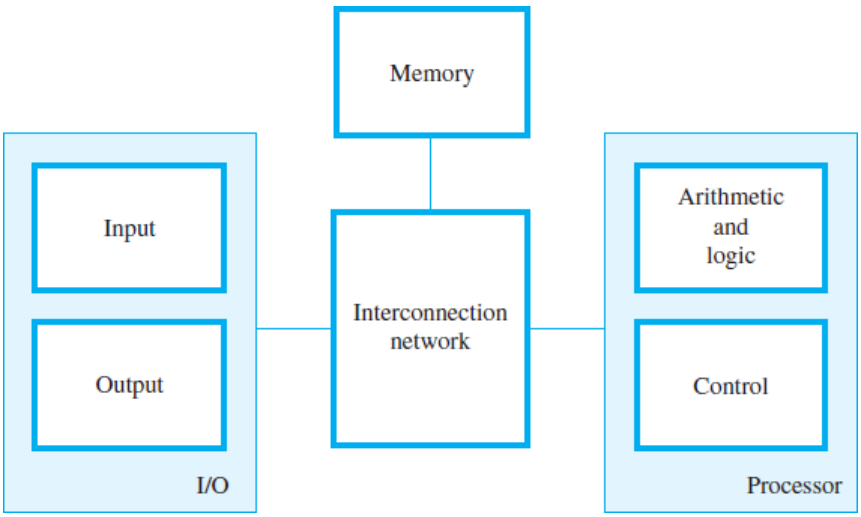
K_B:

A \ BC	00	01	11	10
0	X	X		
1	X	X	1	

$\boxed{K_B = AC}$

Unit-3

1. What are the Basic functional units of a computer? Explain each in detail.



In Input Unit The input unit consists of input devices that are attached to the computer. These devices take input and convert it into binary language that the computer understands. Some of the common input devices are keyboard, mouse, joystick, scanner etc

Central Processing Unit (CPU) Once the information is entered into the computer by the input device, the processor processes it. The CPU is called the brain of the computer because it is the control center of the computer. It first fetches instructions from memory and then interprets them

Arithmetic logic unit: The ALU, as its name suggests performs mathematical calculations and takes logicaldecisions. Arithmetic calculations include addition, subtraction, multiplication and division. Logical decisions involve comparison of two data items to see which one is larger or smaller or equal.

Control Unit : The Control unit coordinates and controls the data flow in and out of CPU and also controls all the operationsof ALU, memory registers and also input/output units. It is also responsible for carrying out all the instructions stored in the program. It decodes the fetched instruction, interprets it and sends control signals to input/output devices until the required operation is done properly by ALU and memory

Memory Registers : A register is a temporary unit of memory in the CPU. These are used to store the data which is directly used by the processor.

Memory : Memory attached to the CPU is used for storage of data and instructions and is called internal memory

Output Unit : The output unit consists of output devices that are attached with the computer

2. Explain in detail about i) Instruction Execution and Straight-Line Sequencing
   ii) Branching

• The program is executed as follows:

1) Initially, the address of the first instruction is loaded into PC (Figure 2.8).

2) Then, the processor control circuits use the information in the PC to fetch and execute instructions, one at a time, in the order ofincreasing addresses. This is called Straight-Line sequencing.

3) During the execution of each instruction, PC is incremented by 4 to point to next instruction.

• There are 2 phases for Instruction Execution:

1) Fetch Phase: The instruction is fetched from the memory-location and placed in the IR.

2) Execute Phase: The contents of IR is examined to determine which operation is to beperformed. The specified-operation is thenperformed by the processor.

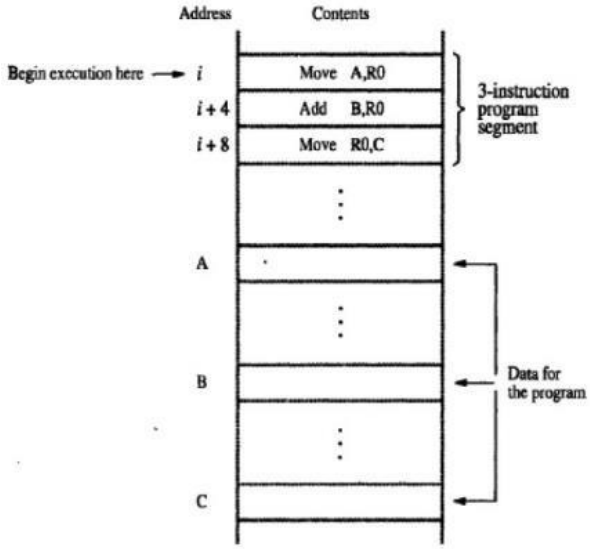
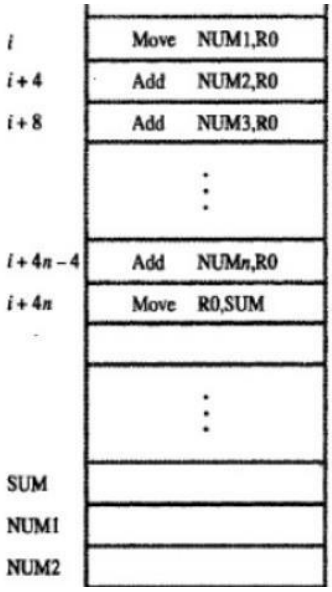


Figure 2.8 A program for C ← [A] + [B].



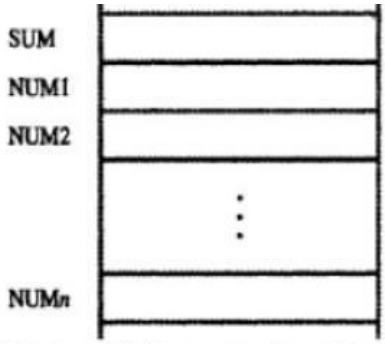


Figure 2.9 A straight-line program for adding *n* numbers.

3. Explain in detail about Addressing Modes.

1) Immediate Mode

• The operand is given explicitly in the instruction.

• For example, the instruction Move #200, R0 ;Place the value 200 in register R0.

2) Register Mode

• The operand is the contents of a register.

• The name (or address) of the register is given in the instruction.

• Registers are used as temporary storage locations where the data in a register are accessed.

• For example, the instruction Move R1, R2 ;Copy content of register R1 into register R2.

3) Absolute (Direct) Mode

• The operand is in a memory-location.

• The address of memory-location is given explicitly in the instruction.

• The absolute mode can represent global variables in the program.

• For example, the instruction Move LOC, R2 ;Copy content of memory-location LOC into register R2.

4) INDIRECTION AND POINTERS

• Instruction does not give the operand or its address explicitly.

• Instead, the instruction provides information from which the new address of the operand can be determined.

Indirect Mode

6 INDEXING AND ARRAYS

- 7 Base with Index Mode
  RELATIVE MODE
  Auto Increment Mode
  Auto Decrement Mode

4. Explain in detail about Instruction and Instruction Sequencing.

INSTRUCTIONS & INSTRUCTION SEQUENCING

The tasks carried out by a computer program consist of a sequence of small steps, suchas adding two numbers, testing for a particularcondition, reading a character from the keyboard, or sending a character to be displayed on a display screen.

A computer must have instructions capable of performing 4 types of operations:

1) Data transfers between the memory and the registers (MOV, PUSH, POP, XCHG).

2) Arithmetic and logic operations on data (ADD, SUB, MUL, DIV, AND, OR, NOT).

3) Program sequencing and control(CALL.RET, LOOP, INT).

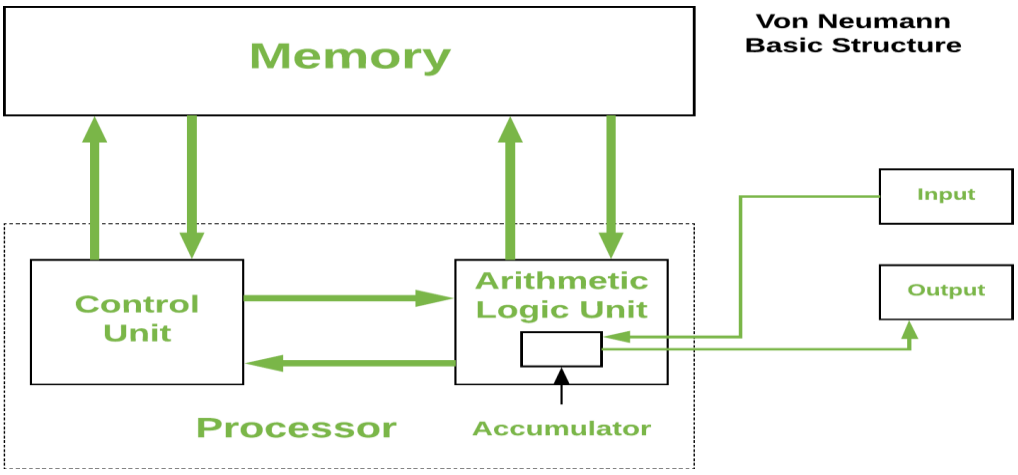
4) I/0 transfers (IN, OUT).

REGISTER TRANSFER NOTATION (RTN)

ASSEMBLY LANGUAGE NOTATION

BASIC INSTRUCTION TYPES

5. Explain in detail about Von Neumann Architecture.

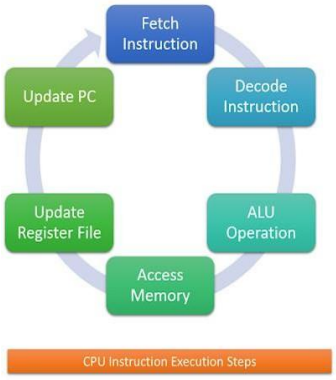


It is also known as **ISA** (Instruction set architecture) computer and is having three basic units:

The Central Processing Unit (CPU)
The Main Memory Unit
The Input/Output Device

Unit-4

1.**Explain in detail about Instruction Execution**.

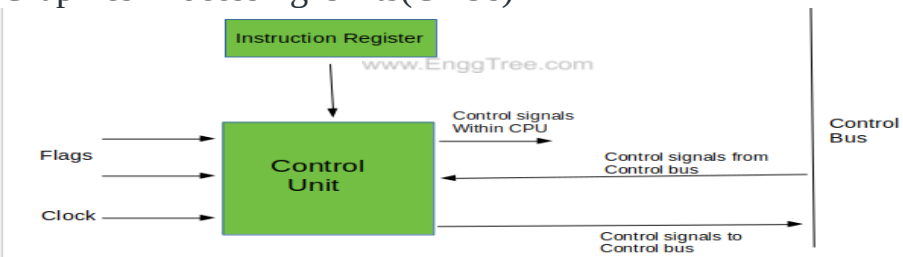


Six steps
Fetch instruction
Decode information
Perform ALU operation
Access memory
Update register file
Update the Program Counter (PC)

2.**Give brief explanation about designing a Control Unit**

Control Unit is the part of the computer's central processing unit (CPU), which directs the operation of the processor. It was included as part of the Von Neumann Architecture by John von Neumann

Control Processing Units(CPUs)
Graphics Processing Units(GPUs)



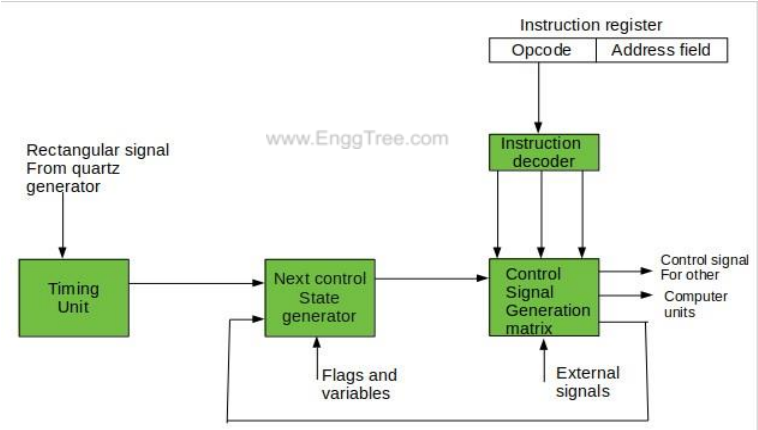
Block Diagram of the Control Unit

Functions of the Control Unit –

It coordinates the sequence of data movements into, out of, and between aprocessor's many sub-units.
It interprets instructions.
It controls data flow inside the processor.
It receives external instructions or commands to which it converts to sequence ofcontrol signals.
It controls many execution units(i.e. ALU, data buffers and registers) containedwithin a CPU.

It also handles multiple tasks, such as fetching, decoding, execution handling andstoring results.

3.Explain in detail about Hardwired Control.



Block diagram of a hardwired control unit of a computer

advantage of Hardwired Control Unit

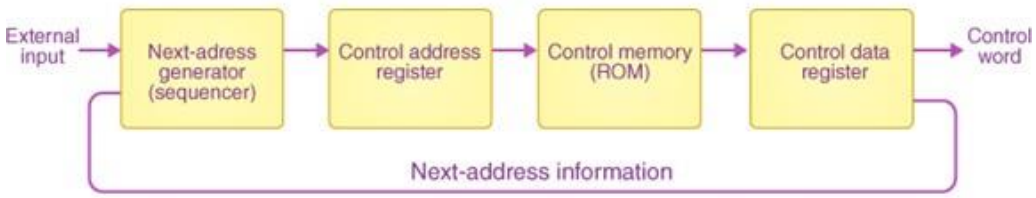
Hardwired Control Unit is quick due to the usage of combinational circuitsto generate signals.
The amount of delay that can occur in the creation of control signals isdependent on the number of gates.
It can be tweaked to get the fastest mode of operation.
Quicker than a micro-programmed control unit.

disadvantage of Hardwired Control Unit

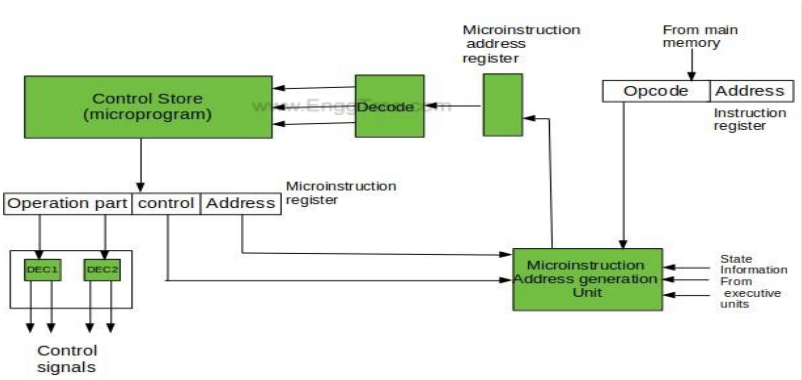
As it require additional control signals to be created, the design becomesmorecomplex (need for more encoders or decoders).
Changes to control signals are challenging since they necessitaterearrangingwires in the hardware circuit.
It's difficult and time-consuming to add a new feature.
It's difficult to evaluate and fix flaws in the initial design**.**

4.Explain in detail about Microprogrammed Control.

Micro program consists of large number of micro instructions. When executing amicro instruction it will generate a appropriate control signal.



Microprogrammed control unit of a Basic Computer



Microprogrammed control unit with a single level control store

ADVANTAGES

It allows for a more methodical control unit design.

It's easier to troubleshoot and modify.

It's more adaptable.

It is used to do complex functions with ease.

DISADVANTAGE

Adaptability comes at a higher price.

It is comparatively slower than a control unit t hat is hardwired.

5.**Explain in detail about Pipelining concepts.**

Pipelining defines the temporal overlapping of processing. Pipelines are emptiness greater than assembly lines in computing that can be used either for instruction processing or, in a more general method, for executing any complex operations

Five phases of instruction execution

Instruction Fetch

Instruction decode

Instruction execution

Memory access

Write back

Pipelining take this instruction execution phases into pipelining stages.

Pipining stages

Instruction Fetch(IF)

Instruction Decode (ID)

Instruction Execution (ALU)

Memory Access (MA)

Write Back(WB)

$$\text{Time between instructions}_{\text{pipelined}} = \frac{\text{Time between instruction}_{\text{nonpipelined}}}{\text{Number of pipe stages}}$$

Types of pipelines

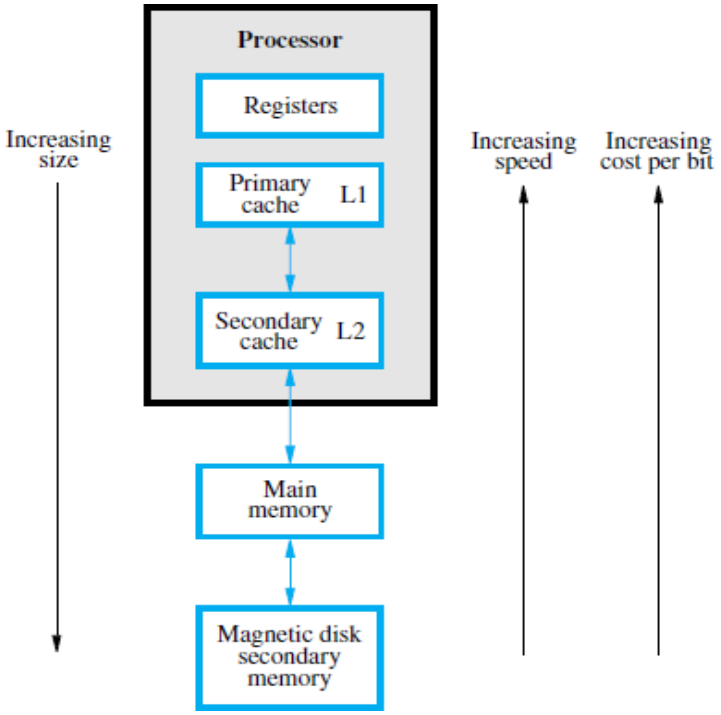
Instruction pipeline

Arithmetic pipeline

UNIT-5

1. Explain in detail about Memory Concepts and Hierarchy

- An ideal memory would be fast, large, and inexpensive. It is clear that a very fast memory can be implemented using static RAM chips. But, these chips are not suitable for implementing large memories, because their basic cells are larger and consume more power than dynamic RAM cells.
- Although dynamic memory units with gigabyte capacities can be implemented at a reasonable cost, the affordable size is still small compared to the demands of large programs with voluminousdata.
- A solution is provided by using secondary storage, mainly magnetic disks, to provide the required memory space. Disks are available at a reasonable cost, and they are used extensively incomputer systems. However, they are much slower than semiconductor memory units.
- In summary, a very large amount of cost-effective storage can be provided by magnetic disks, and a large and considerably faster, yet affordable, main memory can be built with dynamic RAM technology. This leaves the more expensive and much faster static RAM technology to be used in smaller units where speed is of the essence, such as in cache memories.
- All of these different types of memory units are employed effectively in a computer system. The entire computer memory can be viewed as the hierarchy depicted in Figure The fastest access is to data held in processor registers. Therefore, if we consider the registers to be part of the memory hierarchy, then the processor registers are at the top in terms of speed of access. Of course, the registers provide only a minuscule portion of the required memory.



Memory hierarchy

- At the next level of the hierarchy is a relatively small amount of memory that can be implemented directly on the processor chip. This memory, called a processor cache, holds copies of theinstructions and data stored in a much larger memory that is provided externally.
- There are often two or more levels of cache. A primary cache is always located on the processor chip. This cache is small and its access time is comparable to that of processor registers. The primary cache is referred to as the level 1 (L1) cache. A larger, and hence somewhat slower, secondary cache is placed between the primary cache and the rest of the memory. It is referred to as the level 2 (L2) cache. Often, the L2 cache is also housed on the processor chip.
- Some computers have a level 3 (L3) cache of even larger size, in addition to the L1 and L2 caches. An L3 cache, also implemented in SRAM technology, may or may not be on the same chip with the processor and the L1 and L2 caches.

2. **Explain in detail about Memory Management**.

- There are four primary technologies used today in memory hierarchies. Main memory is implemented from DRAM (dynamic random access memory), while levels closer to the processor (caches) use SRAM (static random access memory).
- DRAM is less costly per bit than SRAM, although it is substantially slower. The price difference arises because DRAM uses significantly less area per bit of memory, and DRAMs thus have larger capacity for the same amount of silicon.
- The third technology is flash memory. This nonvolatile memory is the secondary memory in Personal Mobile Devices.
- The fourth technology, used to implement the largest and slowest level in the hierarchy in servers, is magnetic disk. The access time and price per bit vary widely among thesetechnologies, as the table below shows, using typical values for 2012:

Memory technology	Typical access time	$ per GiB in 2012
SRAM semiconductor memory	0.5–2.5 ns	$500–$1000
DRAM semiconductor memory	50–70 ns	$10–$20
Flash semiconductor memory	5,000–50,000 ns	$0.75–$1.00
Magnetic disk	5,000,000–20,000,000 ns	$0.05–$0.10

SRAM Technology

- SRAMs are simply integrated circuits that are memory arrays with (usually) a single access port that can provide either a read or a write.
- SRAMs have a fixed access time to any datum, though the read and write access times may differ.
- SRAMs don't need to refresh and so the access time is very close to the cycle time. SRAMs typically use six to eight transistors per bit to prevent the information from being disturbed when read.
- SRAM needs only minimal power to retain the charge in standby mode.
- Memories that consist of circuits capable of retaining their state as long as power is applied are known as static memories.
- Figure illustrates how a static RAM (SRAM) cell may be implemented. Two inverters are cross-connected to form a latch. The latch is connected to two bit lines by transistors T1 and T2. These transistors act as switches that can be opened or closed under control of the

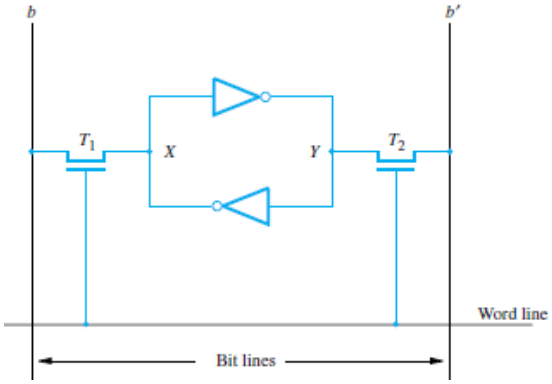
word line. When the word line is at ground level, the transistors are turned off and the latch retains its state.

Read Operation

- In order to read the state of the SRAM cell, the word line is activated to close switches T_1 and T_2. If the cell is in state 1, the signal on bit line b is high and the signal on bit line b is low. The opposite is true if the cell is in state 0. Thus, b and b' are always complements of each other. The Sense/Write circuit at the end of the two bit lines monitors their state and sets the corresponding output accordingly.

Write Operation

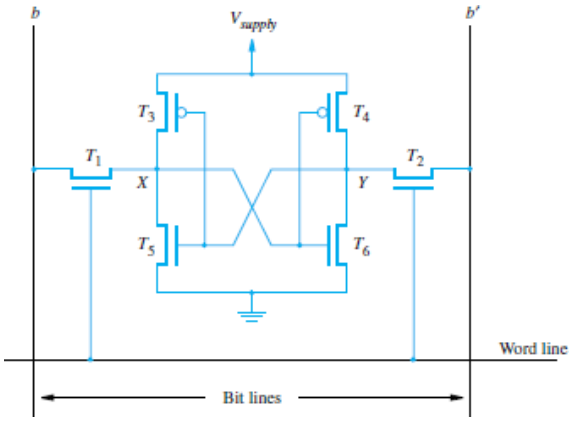
- During a Write operation, the Sense/Write circuit drives bit lines b and b, instead of sensing theirstate. It places the appropriate value on bit line b and its complement on b' and activates the word line. This forces the cell into the corresponding state, which the cell retains when the word line is deactivated.



A static RAM cell

CMOS Cell

- Transistor pairs (T_3, T_5) and (T_4, T_6) form the inverters in the latch . The state of the cell is read or written as just explained. For example, in state 1, the voltage at point X is maintained high byhaving transistors T3 and T6 on, while T4 and T5 are off. If T1 and T2 are turned on, bit lines band b' will have high and low signals, respectively.



An example of a CMOS memory cell

Dynamic RAM:

- Static RAMs are fast, but their cells require several transistors. Less expensive and higher density RAMs can be implemented with simpler cells. But, these simpler cells do not retain theirstate for a long period, unless they are accessed frequently for Read or Write operations. Memories that use such cells are called dynamic RAMs (DRAMs).
- Information is stored in a dynamic memory cell in the form of a charge on a capacitor, but this charge can be maintained for only tens of milliseconds. Since the cell is required to store information for a much longer time, its contents must be periodically refreshed by restoring the capacitor charge to its full value. This occurs when the contents of the cell are read or when newinformation is written into it.
- An example of a dynamic memory cell that consists of a capacitor, C, and a transistor, T, is shown in Figure. To store information in this cell, transistor T is turned on and an appropriatevoltage is applied to the bit line. This causes a known amount of charge to be stored in the capacitor.
- After the transistor is turned off, the charge remains stored in the capacitor, but not for long. Thecapacitor begins to discharge.

3. Explain in detail about Cache Memories.

CACHE MEMORY:

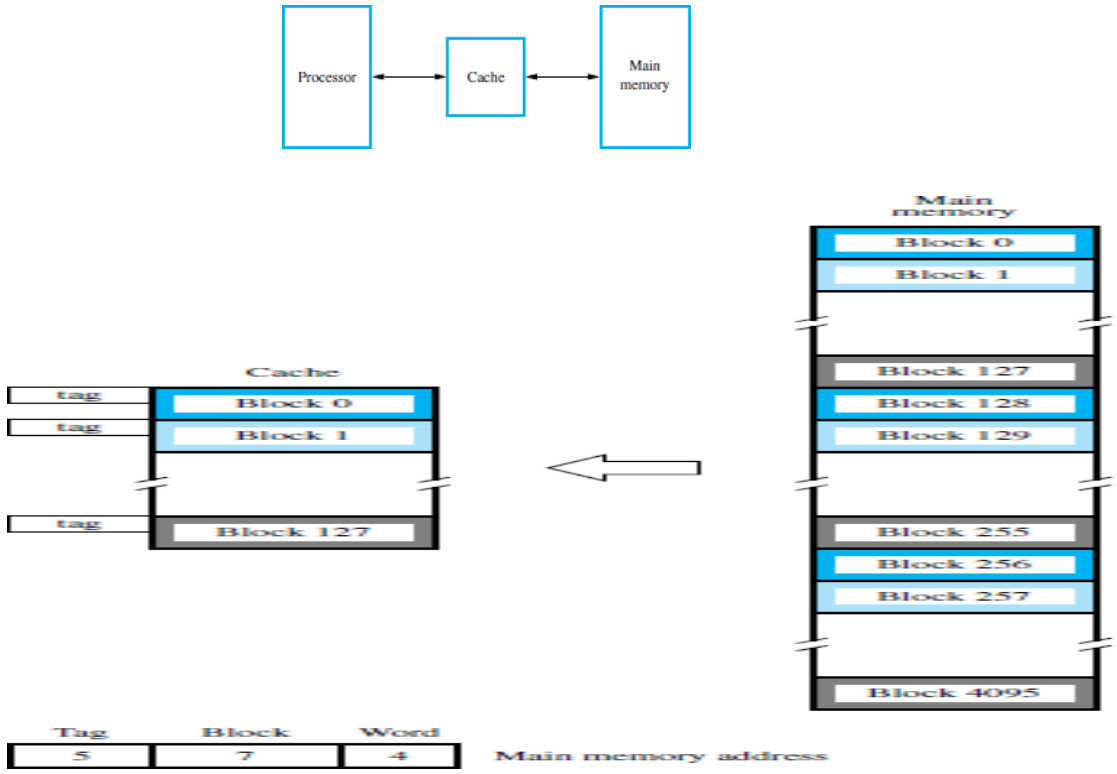
- The cache is a small and very fast memory, interposed between the processor and the main memory. Its purpose is to make the main memory appear to the processor to be much faster than it actually is. The effectiveness of this approach is based on a property of computer programs called locality of reference.
- If programs are executed repeatedly during some time period, this behaviour manifests itself in two ways: temporal and spatial.
- Figure shows such a simple cache, before and after requesting a data item that is not initially in the cache. Before the request, the cache contains a collection of recent referencesX_1, X_2, …, X_{n-1}, and the processor requests a word X_n that is not in the cache. This request results in a miss, and the word X_n is brought from memory into the cache.
- If each word can go in exactly one place in the cache, then it is straightforward to find the word if it is in the cache. The simplest way to assign a location in the cache for each word in memory is to assign the cache location based on the address of the word in memory.
- This cache structure is called **direct mapped**, since each memory location is mapped directly to exactly one location in the cache. The typical mapping between addresses and cache locations for a direct mapped cache is usually simple.
  - For example, almost all direct-mapped caches use this mapping to find a block:(Block address) modulo (Number of blocks in the cache)

The cache just before and just after a reference to a word X_n that is not initially in the cache

- Consider the arrangement in Figure. When the processor issues a Read request, the contents of a block of memory words containing the location specified are transferred into the cache. Subsequently, when the program references any of the locations in this block, the desired contents are read directly from the cache.
- Usually, the cache memory can store a reasonable number of blocks at any given time, but this number is small compared to the total number of blocks in the main memory. The correspondence between the main

memory blocks and those in the cache is specified by a mapping function.

- When the cache is full and a memory word (instruction or data) that is not in the cache is referenced, the cache control hardware must decide which block should be removed to create space for the new block that contains the referenced word. The collection of rules for making this decision constitutes the cache's replacement algorithm.



Use of a cache memory

4. With neat diagram explain in detail about Mapping Techniques.

Mapping Functions

- There are several possible methods for determining where memory blocks are placed in the cache. It is instructive to describe these methods using a specific small example.
- Consider a cache consisting of 128 blocks of 16 words each, for a total of 2048 (2K) words, and assume that the main memory is addressable by a 16-bit address.
- The main memory has 64K words, which we will view as 4K blocks of 16 words each. For simplicity, we have assumed that consecutive addresses refer to consecutive words.

Direct Mapping

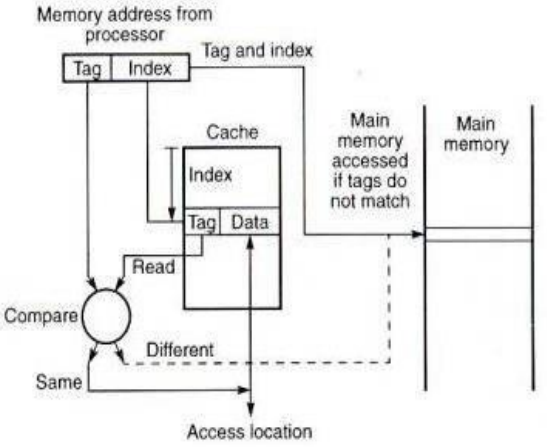
- The simplest way to determine cache locations in which to store memory blocks is the direct-mapping technique. In this technique, block j of the main memory maps onto block j modulo 128 of the cache, as depicted in Figure.
- Thus, whenever one of the main memory blocks 0, 128, 256 . . . are loaded into the cache, it is stored in cache block 0.Blocks 1, 129, 257 . . . are stored in cache block 1, and so on. Since more than one memory block is mapped onto a given cache block position, contention may arise for that position even when the cache is not full.
- For example, instructions of a program may start in block 1 and continue in block 129, possibly after a branch. As this program is executed, both of these blocks must be transferred to the block-1 position in the cache. Contention is resolved by allowing the new block to overwrite the currently resident block.
- With direct mapping, the replacement algorithm is trivial. Placement of a block in the cache is determined by its memory address. The memory address can be divided into three fields, as shown in Figure. The low-order 4 bits select one of 16 words in a block.
- When a new block enters the cache, the 7-bit cache block field determines the cache position in which

this block must be stored. The high-order 5 bits of the memory address of the block are stored in 5 *tag* bits associated with its location in the cache.

- The tag bits identify which of the 32 main memory blocks mapped into this cache position is currently resident in the cache.

Direct-mapped cache

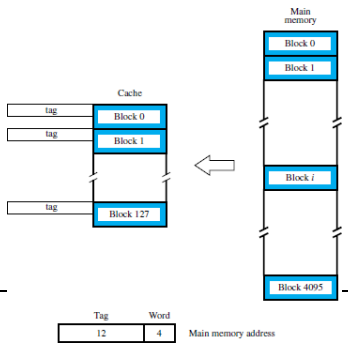
- As execution proceeds, the 7-bit cache block field of each address generated by the processor points to a particular block location in the cache.
- The high-order 5 bits of the address are compared with the tag bits associated with that cache location. If they match, then the desired word is in that block of the cache.
- If there is no match, then the block containing the required word must first be read from the main memory and loaded into the cache. The direct-mapping technique is easy to implement, but it is not very flexible.
- Consider the example shown in figure. The address from the processor is divided in to two fields, a tag and an index. The tag consists of the higher significant bits of the address, which are stored with the data. The index is the lower significant bits of the address used to address the cache.
- When the memory is referenced, the index is first used to access a word in the cache. Then the tag stored in the accessed word is read and compared with the tag in the address. If the two tags are the same, indicating that the word is the one required, access is made to the addressed cache word.
    - If the tags are not the same, indicating that the required word is not in the cache, reference is made to the main memory to find it.



Cache with direct mapping

Associative Mapping

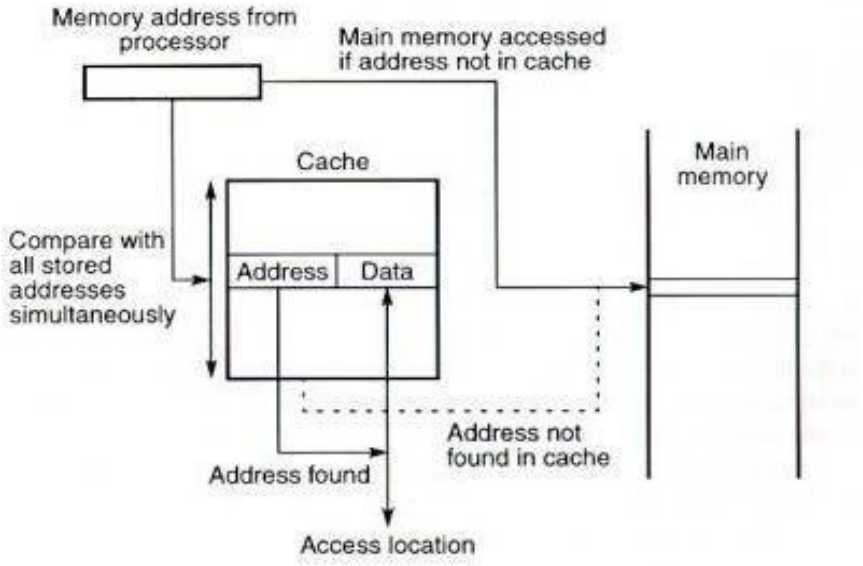
- Figure shows the most flexible mapping method, in which a main memory block can be placed into any cache block position. In this case, 12 tag bits are required to identify a memory block when it is resident in the cache.
- The tag bits of an address received from the processor are compared to the tag bits of each block of the cache to see if the desired block is present. This is called the associative-mapping technique. It gives complete freedom in choosing the cache location in which to place the memory block, resulting in a more efficient use of the space in the cache.



Associative-mapped cache

- When a new block is brought into the cache, it replaces (ejects)an existing block only if the cache is full. In this case, we need an algorithm to select the block to be replaced. Many replacement algorithms are possible.

- The complexity of an associative cache is higher than that of a direct-mapped cache, because of the need to search all 128 tag patterns to determine whether a given block is in the cache. To avoid a long delay, the tags must be searched in parallel. A search of this kind is called an associative search.
- A fully associative cache requires the cache to be composed of associative memory holding both the memory address and the data for each cached line. The incoming memory address is simultaneously compared with all stored addresses using the internal logic of associative memory, as shown in figure.
- If the match is found, the corresponding data is read out. Single words from anywhere within the main memory could be held in cache, if the associative part of the cache is capable of holding a full address.

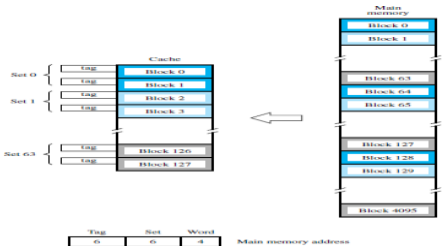


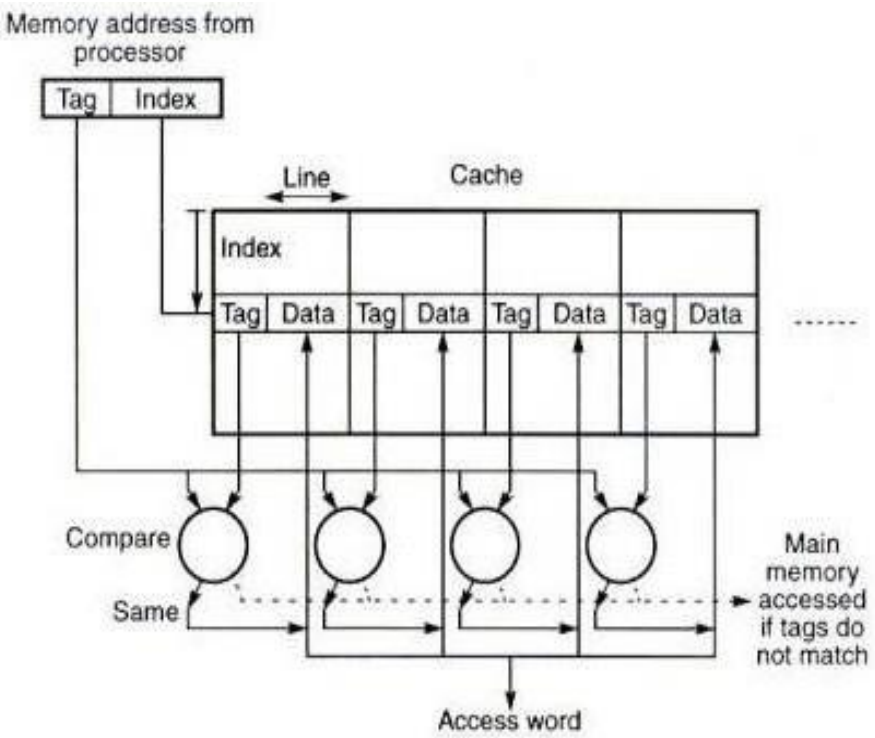
Cache with fully associative mapping

Set-Associative Mapping

- Another approach is to use a combination of the direct- and associative-mapping techniques. The blocks of the cache are grouped into sets, and the mapping allows a block of the main memory to reside in any block of a specific set. Hence, the contention problem of the direct method is eased by having a few choices for block placement. At the same time, the hardware cost is reduced by decreasing the size of the associative search.
- An example of this set-associative-mapping technique is shown in Figure for a cache with two blocks per set. In this case, memory blocks 0, 64, 128,. . . 4032 map into cache set 0, and they can occupy either of the two block positions within this set. Having 64 sets means that the6-bit set field of the address determines which set of the cache might contain the desired block.
- The tag field of the address must then be associatively compared to the tags of the two blocks of the set to check if the desired block is present. This two-way associative search is simple to implement.
- The number of blocks per set is a parameter that can be selected to suit the requirements of a particular computer. For the main memory and cache sizes in Figure, four blocks per set can be accommodated by a 5-bit set field, eight blocks per set by a 4-bit set field, and so on.
- The extreme condition of 128 blocks per set requires no set bits and corresponds to the fully-associative technique, with 12 tag bits. The other extreme of one block per set is the direct-mapping method. A cache that has k blocks per set is referred to as a k-way set-associative

cache.





Set-associative-mapped cache with two blocks per set

Cache with set associative mapping

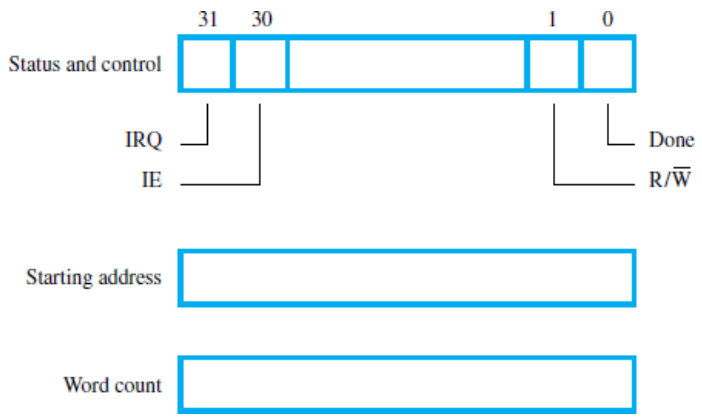
- The cache is divided into sets of blocks. Each block in each set has a stored tag which, together with the index, completes the identification of the block. First, the index of the address from the processor is used to access the set. Then, comparators are used to compare all the tags of the selected set with the incoming tag. If a match is found, the corresponding location is accessed, otherwise, as before; an access to the main memory is made.

5. Explain in detail about DMA.

DIRECT MEMORY ACCESS:

- Blocks of data are often transferred between the main memory and I/O devices such as disks. This section discusses a technique for controlling such transfers without frequent, program-controlled intervention by the processor.
- Data are transferred from an I/O device to the memory by first reading them from the I/O device using an instruction such as Load R2, DATAIN which loads the data into a processor register. Then, the data read are stored into a memory location. The reverse process takes place for transferring data from the memory to an I/O device.
- An instruction to transfer input or output data is executed only after the processor determines that the I/O device is ready, either by polling its status register or by waiting for an interrupt request.

- In either case, considerable overhead is incurred, because several program instructions must be executed involving many memory accesses for each data word transferred. When transferring a block of data, instructions are needed to increment the memory address and keep track of the word count.
- The use of interrupts involves operating system routines which incur additional overhead to save and restore processor registers, the program counter, and other state information.
- An alternative approach is used to transfer blocks of data directly between the main memory and I/O devices, such as disks.
- A special control unit is provided to manage the transfer, without continuous intervention by the processor. This approach is called direct memory access, or DMA. The unit that controls DMAtransfers is referred to as a DMA controller. It may be part of the I/O device interface, or it may be a separate unit shared by a number of I/O devices.
- The DMA controller performs the functions that would normally be carried out by the processor when accessing the main memory.
- For each word transferred, it provides the memory address and generates all the control signals needed. It increments the memory address for successive words and keeps track of the number of transfers.
- Although a DMA controller transfers data without intervention by the processor, its operation must be under the control of a program executed by the processor, usually an operating system routine.
- To initiate the transfer of a block of words, the processor sends to the DMA controller the starting address, the number of words in the block, and the direction of the transfer.
- The DMA controller then proceeds to perform the requested operation. When the entire block has been transferred, it informs the processor by raising an interrupt.
- Figure shows an example of the DMA controller registers that are accessed by the processor to initiate data transfer operations. Two registers are used for storing the starting address and the word count. The third register contains status and control flags. The R/W bit determines the direction of the transfer.
- When this bit is set to 1 by a program instruction, the controller performs a Read operation, that is, it transfers data from the memory to the I/O device. Otherwise, it performs a Write operation. Additional information is also transferred as may be required by the I/O device.
- For example, in the case of a disk, the processor provides the disk controller with information to identify where the data is located on the disk.



Typical registers in a DMA controller

- When the controller has completed transferring a block of data and is ready to receive another command, it sets the Done flag to 1. Bit 30 is the Interrupt-enable flag, IE.
- When this flag is set to 1, it causes the controller to raise an interrupt after it has completed

transferring a block of data. Finally, the controller sets the IRQ bit to 1 when it has requested an interrupt.

- Figure shows how DMA controllers may be used in a computer system. One DMA controller connects a high-speed Ethernet to the computer's I/O bus.
- The disk controller, which controls two disks, also has DMA capability and provides two DMA channels. It can perform two independent DMA operations, as if each disk had its own DMA controller.
- The registers needed to store the memory address, the word count, and so on, are duplicated, so that one set can be used with each disk.

6. Explain in detail virtual memory

VIRTUAL MEMORY:

- In most modern computer systems, the physical main memory is not as large as the address space of the processor. For example, a processor that issues 32-bit addresses has an addressable space of 4G bytes. The size of the main memory in a typical computer with a 32-bit processor may range from 1G to 4G bytes.
- If a program does not completely fit into the main memory, the parts of it not currently being executed are stored on a secondary storage device, typically a magnetic disk. As these parts are needed for execution, they must first be brought into the main memory, possibly replacing other parts that are already in the memory. These actions are performed automatically by the operatingsystem, using a scheme known as virtual memory.
- Application programmers need not be aware of the limitations imposed by the available main memory. They prepare programs using the entire address space of the processor.
- Under a virtual memory system, programs, and hence the processor, reference instructions and data in an address space that is independent of the available physical main memory space. The binary addresses that the processor issues for either instructions or data are called virtual or logical addresses. These addresses are translated into physical addresses by a combination of hardware and software actions.
- If a virtual address refers to a part of the program or data space that is currently in the physical memory, then the contents of the appropriate location in the main memory are accessed immediately. Otherwise, the contents of the referenced address must be brought into a suitable location in the memory before they can be used.
- Figure shows a typical organization that implements virtual memory. A special hardware unit, called the Memory Management Unit (MMU), keeps track of which parts of the virtual address space are in the physical memory.

When the desired data or instructions are in the main memory, the MMU translates the virtual address into the corresponding physical address. Then, the requested memory access proceeds in the usual manner. If the data are not in the main memory, the MMU causes the operating system to transfer the data from the disk to the memory

7. Explain in detail USB.

- The Universal Serial Bus (USB) is the most widely used interconnection standard. A large variety of devices are available with a USB connector, including mice, memory keys, disk drives, printers, cameras, and many more.
- The commercial success of the USB is due to its simplicity and low cost. The original USB specification supports two speeds of operati$_{1}$o$_{4}$n$_{4}$, called low-speed (1.5 Megabits/s) and full-speed

(12 Megabits/s). Later, USB 2, called High-Speed USB, was introduced. It enables data transfersat speeds up to 480 Megabits/s.

- As I/O devices continued to evolve with even higher speed requirements, USB 3 (called Superspeed) was developed. It supports data transfer rates up to 5 Gigabits/s.
- The USB has been designed to meet several key objectives:
  - Provide a simple, low-cost, and easy to use interconnection system
  - Accommodate a wide range of I/O devices and bit rates, including Internet connections,and audio and video applications
  - Enhance user convenience through a "plug-and-play" mode of operation

Device Characteristics

- The kinds of devices that may be connected to a computer cover a wide range of functionality. The speed, volume, and timing constraints associated with data transfers to and from these devices vary significantly.
- In the case of a keyboard, one byte of data is generated every time a key is pressed, which may happen at any time. These data should be transferred to the computer promptly. Since the event of pressing a key is not synchronized to any other event in a computer system, the data generated by the keyboard are called asynchronous.
- Furthermore, the rate at which the data are generated is quite low. It is limited by the speed of the human operator to about 10 bytes per second, which is less than 100 bits per second.
- Consider now a different source of data. Many computers have a microphone, either externally attached or built in. The sound picked up by the microphone produces an analog electrical signal, which must be converted into a digital form before it can be handled by the computer. This is accomplished by sampling the analog signal periodically.
- For each sample, an analog-to-digital (A/D) converter generates an *n*-bit number representing the magnitude of the sample. The number of bits, *n*, is selected based on the desired precision with which to represent each sample.
- Later, when these data are sent to a speaker, a digital to- analog (D/A) converter is used to restore the original analog signal from the digital format. A similar approach is used with video information from a camera.
- The sampling process yields a continuous stream of digitized samples that arrive at regular intervals, synchronized with the sampling clock. Such a data stream is called isochronous, meaning that successive events are separated by equal periods of time. A signal must be sampled quickly enough to track its highest-frequency components.
- An important requirement in dealing with sampled voice or music is to maintain precise timing in the sampling and replay processes. A high degree of jitter (variability in sample timing) is unacceptable. Hence, the data transfer mechanism between a computer and a music system must maintain consistent delays from one sample to the next. Otherwise, complex buffering and retiming circuitry would be needed.
- Data transfers for images and video have similar requirements, but require much higher data transfer rates.
- To maintain the picture quality of commercial television, an image should be represented by about 160 kilobytes and transmitted 30 times per second. Together with control information, this yields a total bit rate of 44 Megabits/s. Higher-quality images, as in HDTV (High Definition TV), require higher rates.
- Large storage devices such as magnetic and optical disks present different requirements. Their connection to the computer requires a data transfer bandwidth of at least 40 or 50 Megabits/s.