CBCS SCHEME

USN 1 C R 2 9 C S 0 6 5

BCS302

Third Semester B.E/B.Tech. Degree Examination, Dec.2025/Jan.2026
Digital Design and Computer Organization

Time: 3 hrs. Max. Marks:100

Note: *1. Answer any FIVE full questions, choosing ONE full question from each module.*
2. M : Marks , L: Bloom's level , C: Course outcomes.

		Module – 1	M	L	C
1	a.	Obtain the minimum expression for the POS expression : $F(A, B, C, D) = \pi M(0, 1, 5, 7, 9, 13, 15) + d(3, 10)$.	5	L2	CO1
	b.	Implement the following logic function in SOP form using NOR gates. $Y = A\bar{B} + B\bar{C} + ABC$.	5	L3	CO1
	c.	Identify the essential prime implicants of the following functions : $F(w, x, y, z) = (0, 1, 4, 5, 6, 7, 9, 11, 14, 15)$ $F(A, B, C, D) = (0, 2, 3, 5, 7, 8, 10, 11, 14, 15)$.	10	L3	CO1
		OR			
2	a.	Demonstrate the positive and negative logic using AND gate.	5	L2	CO1
	b.	Simplify the following Boolean functions using K-map : i) $F(P, Q, R, S) = \Sigma(0, 2, 5, 7, 8, 10, 13) + d(1, 4, 15)$ ii) $F(A, B, C, D) = (\bar{A} + B + C)(\bar{A} + \bar{C} + D)(\bar{B} + C + D)$.	10	L3	CO1
	c.	Explain Dataflow Modeling in verilogy with an example program.	5	L1	CO1
		Module – 2			
3	a.	Explain the difference between combinational and sequential circuits with their block diagrams and examples.	5	L2	CO2
	b.	Write the verilog program to implement full adder and full subtractor circuits.	7	L2	CO2
	c.	Describe and explain 4 bit adder with carry look ahead.	8	L3	CO2
		OR			
4	a.	Implement the Boolean function : $F(A, B, C, D) = \Sigma m(1, 3, 4, 11, 12, 13, 14, 15)$ using 8 : 1 MUX.	5	L3	CO2
	b.	What is encoder? Design 8 : 3 encoder circuits with logic diagram and truth table and also list its applications.	7	L3	CO2
	c.	What is Latch? Demonstrate the working of SR flip-flop and D Flip-flop and write the characteristics table and equations.	8	L3	CO2

		Module 3			
5	a.	What do you mean by an addressing mode? Explain any 5 addressing modes.	10	L2	CO3
	b.	Describe the Big-endian and Little-endian address assignment.	5	L1	CO3
	c.	A program with 5000 machine instructions needs an average of 3 basic steps to execute one instruction. Find the performance of the computer having a clock speed of 500 KHz.	5	L3	CO3
		OR			
6	a.	Demonstrate the Branching operations using loop to add n numbers with block diagram.	8	L3	CO3
	b.	Show how below expression will be executed in one address and three address processor in accumulator organization. X = (A * B) + (C * D).	7	L3	CO3
	c.	What are Condition Code Flags? Mention the significance of the flag N, Z, V and C.	5	L1	CO3
		Module – 4			
7	a.	Explain memory mapped I/O and I/O interface for an input device with a diagram.	10	L2	CO4
	b.	Explain DMA with a neat diagram.	10	L4	CO4
		OR			
8	a.	Explain how to handle interrupt from multiple devices using daisy chain and priority scheme.	10	L3	CO4
	b.	Explain centralized and distributed Bus Arbitration approaches.	10	L2	CO4
		Module – 5			
9	a.	With a diagram, explain the single bus organization of the data path inside a processor.	10	L2	CO5
	b.	Describe the basic idea of instruction pipeline.	10	L2	CO5
		OR			
10	a.	Explain the process of fetching word from memory in processor.	10	L4	CO5
	b.	Explain the pipeline performance of a processor and pipeline stalls.	10	L2	CO5

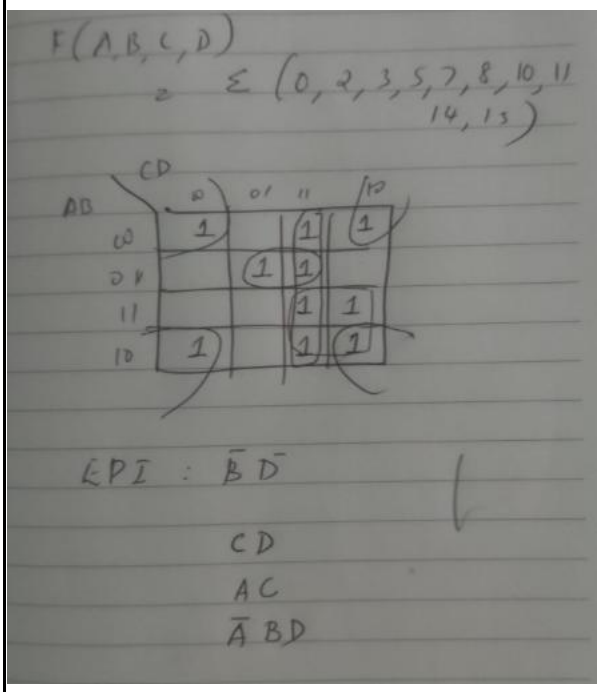
* * * * *

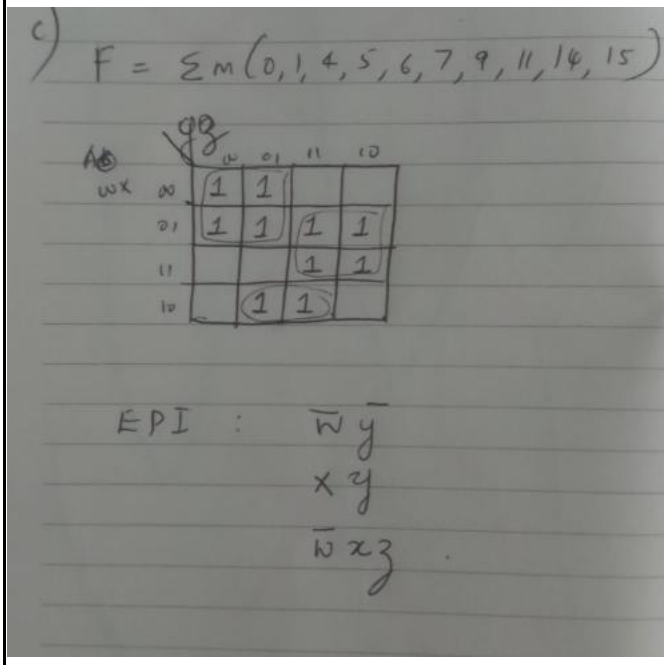
VTU QUESTION PAPER SOLUTION
FEB 2026

Sub:	Digital Design and Computer Organization					Sub Code:	BCS302	Branch:	CSE	
Date:	4/1/2026	Duration:	3hrs	Max Marks:	100	Sem / Sec:	III / A, B, C		OBE	

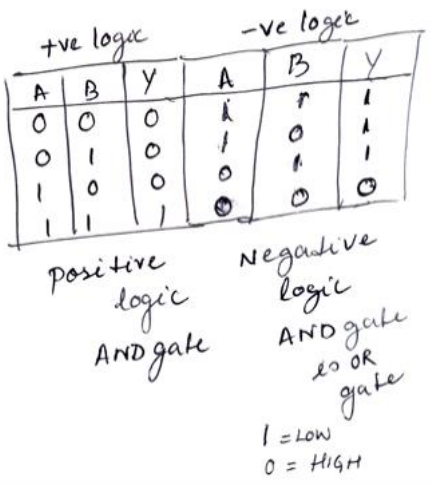
Answer any FIVE FULL Questions

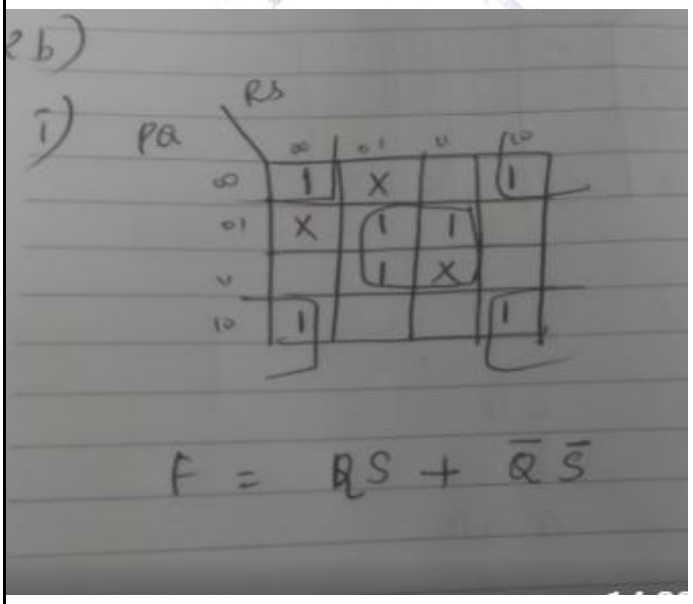
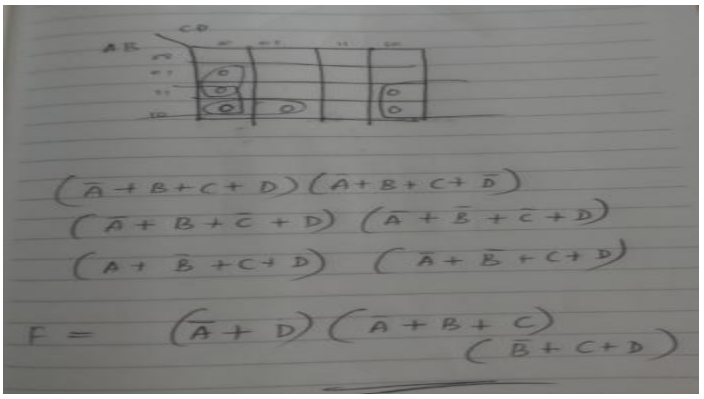
		MARKS	CO	RBT
1a	Obtain the minimum expression for the POS expression : $F(A, B, C, D) = \pi M(0, 1, 5, 7, 9, 13, 15) + d(3, 10)$.	[5]	1	L2
1b	Implement the following logic function in SOP form using NOR gates. $Y = A\bar{B} + B\bar{C} + ABC$.	[5]	1	L3
1c	Identify the essential prime implicants of the following functions : $F(w, x, y, z) = (0, 1, 4, 5, 6, 7, 9, 11, 14, 15)$ $F(A, B, C, D) = (0, 2, 3, 5, 7, 8, 10, 11, 14, 15)$.	[10]	1	L3



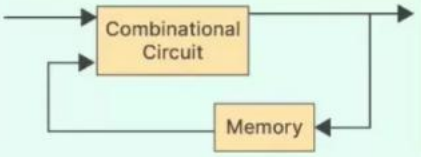


2a	Demonstrate the positive and negative logic using AND gate.	[5]	1	L2



2b	Simplify the following Boolean functions using K-map : i) $F(P, Q, R, S) = \Sigma(0, 2, 5, 7, 8, 10, 13) + d(1, 4, 15)$ ii) $F(A, B, C, D) = (\overline{A}+B+C)(\overline{A}+\overline{C}+D)(\overline{B}+C+D)$. 	[10]	1	L3
2c	 Explain Dataflow Modeling in verilogy with an example program. **module ldataflow(a, b,y0,y1,y2,y3,y4,y5,y6); input a,b; output y0,y1,y2,y3,y4,y5,y6; assign y0=!a; assign y1=a&b; assign y2=a|b; assign y3=!(a&b); assign y4=!(a|b); assign y5=a^b; assign y6=!(a^b); endmodule**	[5]	1	L1

3a	Explain the difference between combinational and sequential circuits with heir block diagrams and examples.	[5]	2	L2

Combinational Circuit	Sequential Circuit
Output only depends on the present input	Output depends on present input & past output
Memory element is absent	Memory element is present
No clock signal is applied	Clock signal is required
	
Example - Half Adder, Full Adder, Multiplexer	Example - Flipflop, Counters, Registers

3b	Write the verilog program to implement full adder and full subtractor circuits.	[7]	2	L2

Verilog code:

Half Adder-	**Half Substractor-**

module halfadder(a,b,sum,carry);
input a,b;
output sum,carry;
assign sum=a^b;
assign carry=a&b;
endmodule
	module HS(a,b,diff,borrow);
	input a,b;
	output diff,borrow;
	assign diff=a^b;
	assign borrow=!(a)&b;
	endmodule

Full Adder-	**Full Substractor-**

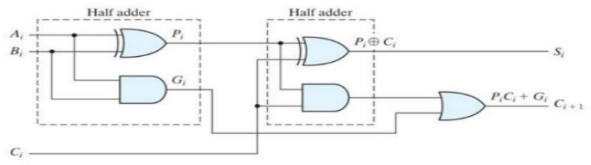
module FA(a,b,c,sum,carry);
input a,b,c;
output sum,carry;
assign sum=a^b^c;
assign carry=(a&b)||(b&c)||(c&a);
endmodule
	module FS(a,b,c,diff,borrow);
	input a,b,c;
	output diff,borrow;
	assign diff=a^b^c;
	assign borrow=((!a)&b)|(b&c)|((!a)&c);
	endmodule

3c	Describe and explain4-bit adder with carry look ahead?	[8]	2	L3

Input carry C_3 does not settle to its final value until C_3 is available from the previous stage.

Similarly, C_2 has to wait for C_1 and so on down to C_0. Thus, only after the carry propagates and ripples through all stages will the last output S_3 and carry C_4 settle to their final correct value.

The problem faced by a 4 bit ripple adder is the delay There are several techniques for reducing the carry propagation time in a parallel adder. The most widely used technique employs the principle of carry lookahead logic .



Consider the circuit of the full adder s in If we define two new binary variables

Pi = Ai ⊕ B i G i = Ai B i

- The output sum and carry can respectively be expressed as

propagate into stage i + 1 (i.e., whether an assertion of C i will propagate to an assertion of C i + 1).

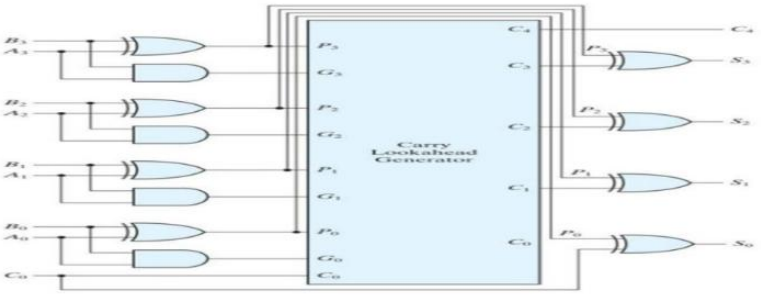
$$= G1 + P1(G0 + P0C0)$$
$$= G1 + P1G0 + P1P0C0$$

)

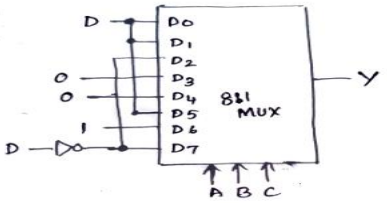
$$C3 = G2 + P2C2$$
$$= G2 + P2G1 + P2P1G0 + P2P1P0C0$$

Since the carry of the different stages depends on only CO we are able to execute the addition in a much faster way.

Circuit of carry look ahead adder:



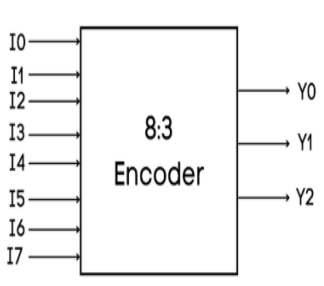
4a Implement the Boolean function :
F(A, B, C, D) = Σm(1, 3, 4, 11, 12, 13, 14, 15) using 8 : 1 MUX. [5] 2 L3

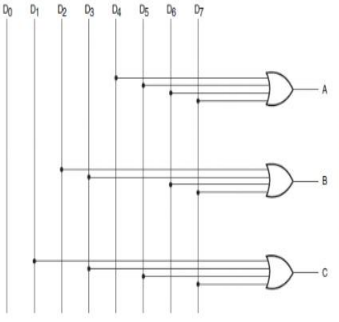


$F(A,B,C,D) = \Sigma m\,(1,3,4,11,12,13,14,15)$

	A	B	C	D	Y		
0	0	0	0	0	0	D	D0
1	0	0	0	1	1		
2	0	0	1	0	0	D	D1
3	0	0	1	1	1		
4	0	1	0	0	1	$\bar{D}$	D2
5	0	1	0	1	0		
6	0	1	1	0	0	0	D3
7	0	1	1	1	0		
8	1	0	0	0	0	0	D4
9	1	0	0	1	0		
10	1	0	1	0	0	D	D5
11	1	0	1	1	1		
12	1	1	0	0	1	1	D6
13	1	1	0	1	1		
14	1	1	1	0	1	$\bar{D}$	D7
15	1	1	1	1	0		

4b What is encoder? Design 8 : 3 encoder circuits with logic diagram and truth table and also list its applications. [7] 2 L3



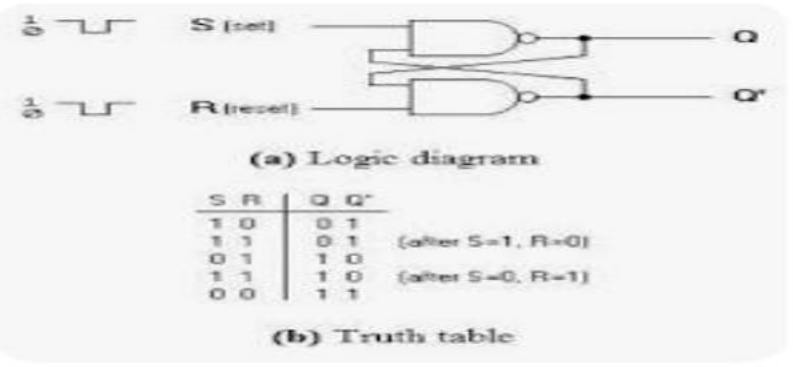


D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Applications: used in scenarios where a large number of input signals need to be represented by a smaller number of output bits.

4c	What is Latch? Demonstrate the working of SR flip-flop and D Flip-flop and write the characteristics table and equations.	[8]	2	L3

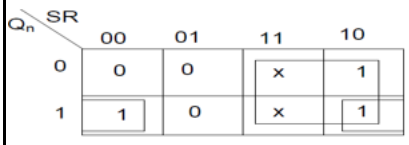
Positive edges occur at t_0, t_1, t_2, t_3 and t_4.
At t_0, S=0 and R=0, hence no change in the output and Q=0.
At t_1, S=1 and R=0, hence the output is set and Q=1.
At t_2, S=0 and R=1, hence the output is reset and Q=0.
At t_3, S=1 and R=0, hence the output is set and Q=1
At t_4, S=0 and R=0, hence no change in the output and Q=1.



(a) Logic diagram

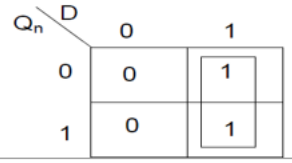
S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after S=1, R=0)
0	1	1	0	
1	1	1	0	(after S=0, R=1)
0	0	1	1	

(b) Truth table



Characteristic Equation for SR flip flop is

$Q_{n+1}=S+\overline{R}Q_n$



Characteristic Equation for D flip flop is

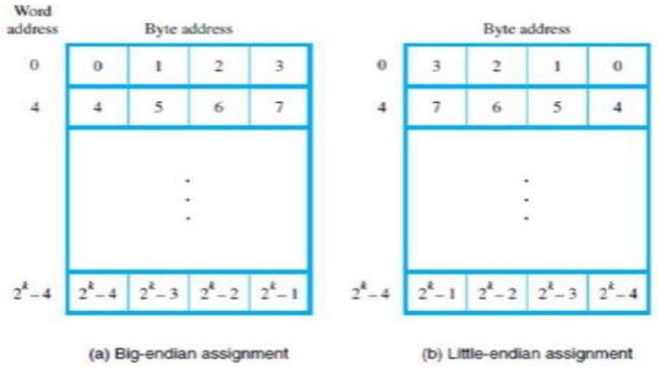
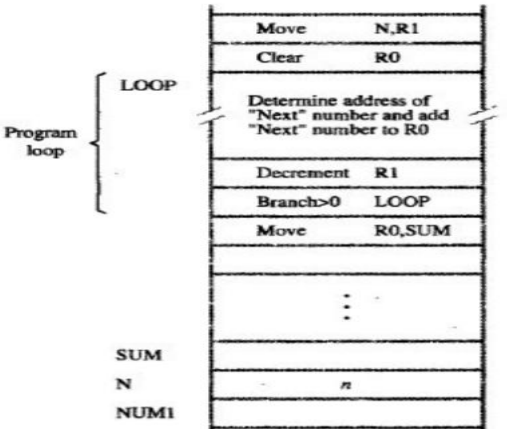
$Q_{n+1}=D$

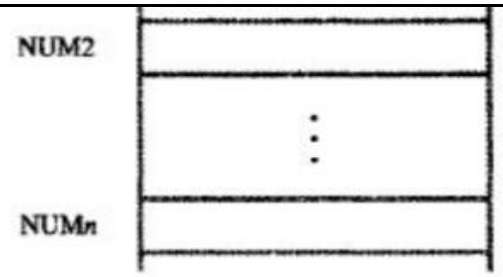
5a	What do you mean by an addressing mode? Explain any 5 addressing modes.	[10]	3	L2

Name	Assembler syntax	Addressing function
Immediate	#Value	Operand = Value
Register	R_i	EA = R_i
Absolute (Direct)	LOC	EA = LOC
Indirect	(R_i) (LOC)	EA = $[R_i]$ EA = [LOC]
Index	$X(R_i)$	EA = $[R_i]$ + X
Base with index	(R_i,R_j)	EA = $[R_i]$ + $[R_j]$
Base with index and offset	$X(R_i,R_j)$	EA = $[R_i]$ + $[R_j]$ + X
Relative	X(PC)	EA = [PC] + X
Autoincrement	(R_i)+	EA = $[R_i]$; Increment R_i
Autodecrement	−(R_i)	Decrement R_i; EA = $[R_i]$

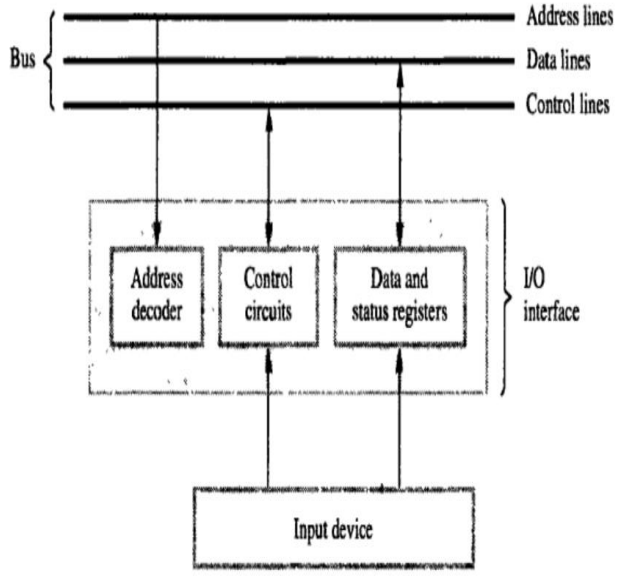
EA = effective address
Value = a signed number

5b	Describe the Big-endian and Little-endian address assignment.	[5]	3	L1

5c	**BIG-ENDIAN & LITTLE-ENDIAN ASSIGNMENTS** • There are two ways in which byte-addresses are arranged (Figure 2.3). **1) Big-Endian:** Lower byte-addresses are used for the more significant bytes of the word. **2) Little-Endian:** Lower byte-addresses are used for the less significant bytes of the word • In both cases, byte-addresses 0, 4, 8. . . . . are taken as the addresses of successive words in the memory.  (a) Big-endian assignment (b) Little-endian assignment A program with 5000 machine instructions needs an average of 3 basic steps to execute one instruction. Find the performance of the computer having a clock speed of 500 KHz. • Number of instructions = **5000** • Average basic steps per instruction = **3** • Clock speed = **500 kHz = 500,000 cycles/sec** **Step 2: Time per clock cycle** $$\text{Clock period} = \frac{1}{\text{clock frequency}} = \frac{1}{500000}$$ $$= 2 \times 10^{-6} \text{ sec} = 2\,\mu s$$ **Step 3: Total execution time** $$\text{Execution time} = 15000 \times 2 \times 10^{-6}$$ $$= 30000 \times 10^{-6}$$ $$= 0.03 \text{ sec}$$	[5]	3	L3
6a	Demonstrate the Branching operations using loop to add n numbers with block diagram.  	[8]	3	L3

6b	 Show how below expression will be executed in one address and three address processor in accumulator organization. X = (A * B) + (C * D). ONE ADDRESS ``` LOAD A MUL B STORE T LOAD C MUL D ADD T STORE X ``` THREE ADDRESS ``` MUL T1, A, B MUL T2, C, D ADD X, T1, T2 ``` What are Condition Code Flags? Mention the significance of the flag N, Z, V and C. ***CONDITION CODE BITS*** • The processor consists of series of flip-flops to store the status information after ALU operation. • It keeps track of the results of various operations, for subsequent usage. • The series of flip-flip-flops used to store the status and control information of the processor is called as "Condition Code Register". It defines 4 flags. The format of condition code register is as follows. | C | V | Z | N | 1 N (NEGATIVE) Flag: It is designed to differentiate between positive and negative result. It is set 1 if the result is negative, and set to 0 if result is positive. 2 Z (ZERO) Flag: It is set to 1 when the result of an ALU operation is found to zero, otherwise it is cleared. 3 V (OVER FLOW) Flag: In case of 2^s Complement number system n-bit number is capable of representing a range of numbers and is given by -2^{n-1} to $+2^{n-1}$. The Over-Flow flag is set to 1 if the result is found to be out of this range. 4 C (CARRY) Flag : This flag is set to 1 if there is a carry from addition or borrow from subtraction, otherwise it is cleared.	[8] [5]	3 3	L3 L1
7a	Explain memory mapped I/O and I/O interface for an input device with a diagram.	[10]	4	L2

- In this technique, a separate bus is used for I/O devices and memory to transfer the data to CPU. Address space for memory and I/O devices are different.
- Hence two sets of instruction are used for data transfer.
- One set for memory operations and another set for I/O operations.Whole address space is available for the program.



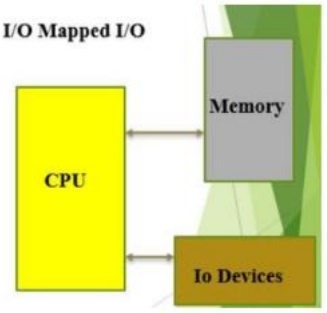
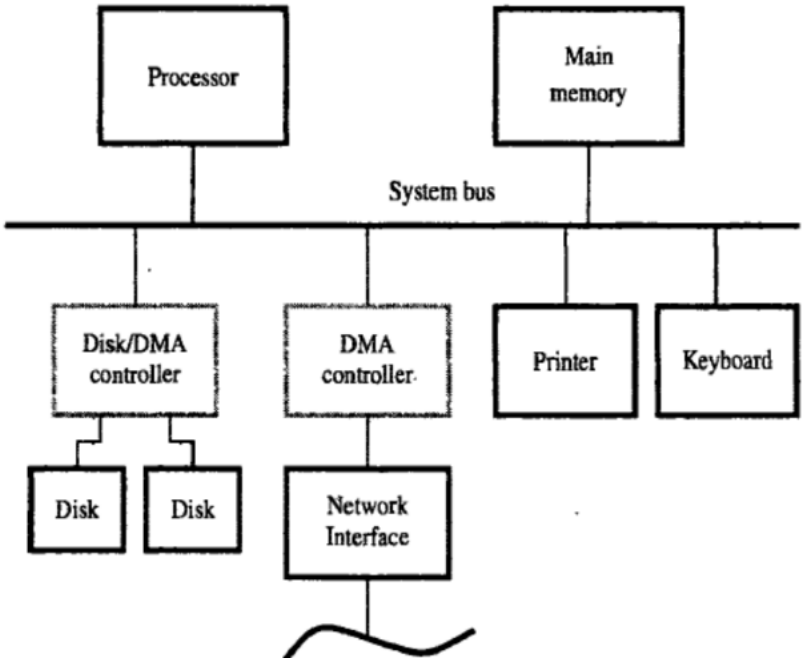


Fig: I/O interface for an input device

7b

Explain DMA with a neat diagram.

[10] 4 L4



- The DMA controller connects two external devices namely disk 1 and disk 2 to system bus asshown in the above fig.
- The DMA controller also interconnects high speed network devices to system bus as shownin the above fig.
- Let us consider direct data transfer operation by means of DMA controller without the involvement of CPU in between main memory and disk 1 as indicated by dotted lines (in the fig.).

- To establish direct data transfer operation between main memory and disk
  1. DMA controller request the processor to obtain **3 parameters** namely:
     1) **Starting address of the memory block.**
     2) **No of words to be transferred.**
     3) **Type of operation (Read or Write).**

8a Explain how to handle interrupt from multiple devices using daisy chain and priority scheme. [10] 4 L3

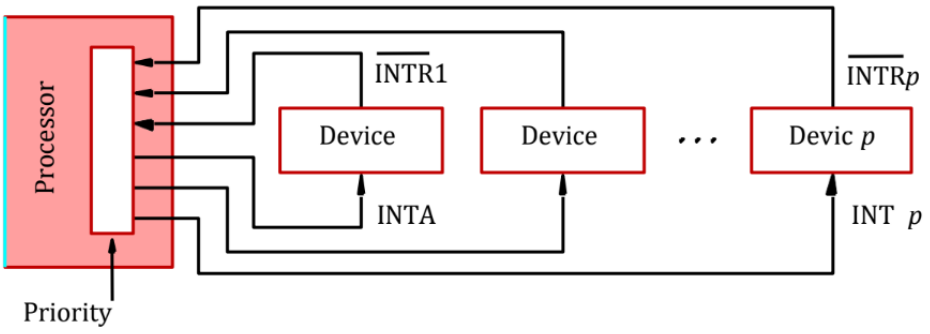
HANDLING MULTIPLE DEVICES

While handling multiple devices, the issues concerned are:

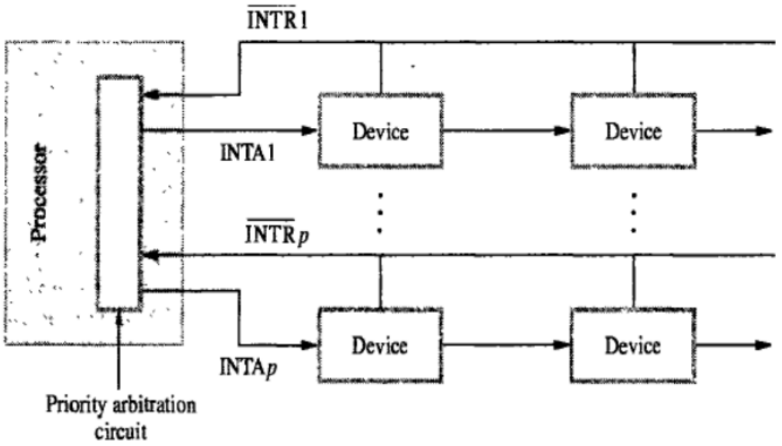
- How can the processor recognize the device requesting an interrupt?
- How can the processor obtain the starting address of the appropriate ISR?
- Should a device be allowed to interrupt the processor while another interrupt isbeing serviced?
- How should 2 or more simultaneous interrupt-requests be handled?

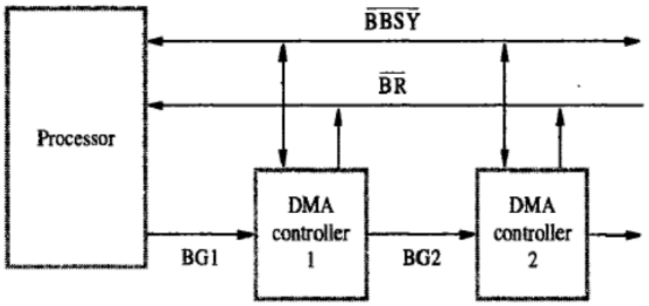
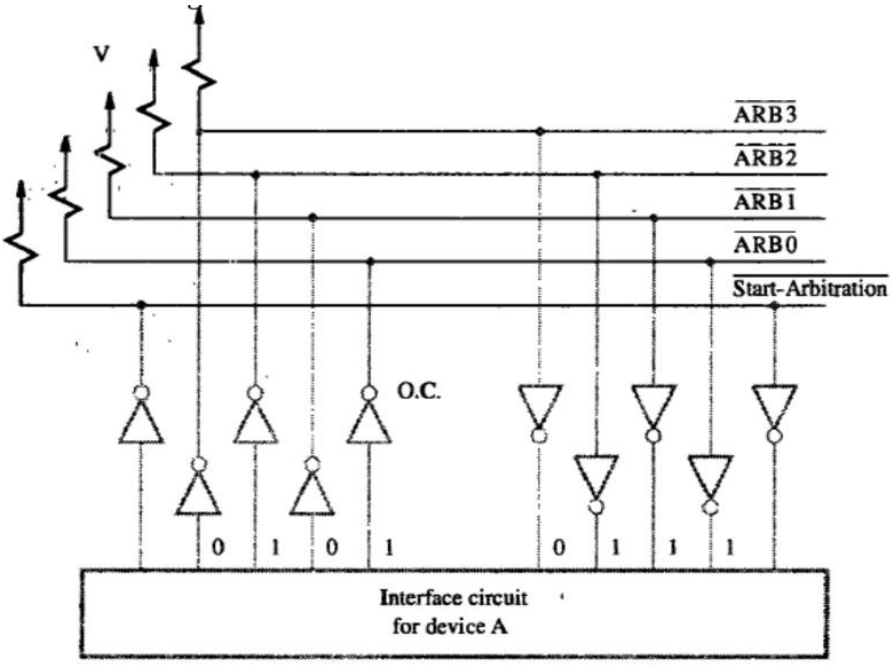
VECTORED INTERRUPT

- A device requesting an interrupt identifies itself by sending a special-code to processor over bus.
- Then, the processor starts executing the ISR.
- The special-code indicates starting-address of ISR.
- The special-code length ranges from 4 to 8 bits.
- The location pointed to by the interrupting-device is used to store the staring address to ISR.

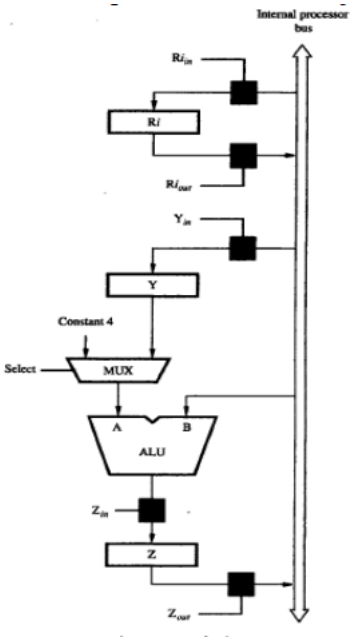


ARRANGEMENT OF PRIORITY GROUPS



8b	Explain centralized and distributed Bus Arbitration approaches. a) **Centralized bus arbitration:** In this technique CPU acts as a bus-master or any control unit connected to bus can be acts as a busmaster.  	[10]	4	L2
9a	With a diagram, explain the single bus organization of the data path inside a processor. • Here the processor contain only a single bus for the movement of data, address and instructions. • ALU and all the registers are interconnected via a **Single Common Bus** (Figure 7.1). • Data & address lines of the external **memory-bus** is connected to the internalprocessor-bus via MDR & MAR respectively. (MDR -> Memory Data Register, MAR -> Memory Address Register). • **MDR** has 2 inputs and 2 outputs. Data may be loaded → into MDR either from memory-bus (external) or → from processor-bus (internal). • **MAR**"s input is connected to internal-bus; MAR"s output is connected to external- bus. (address sent from processor to memory only)	[10]	5	L2

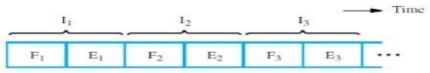
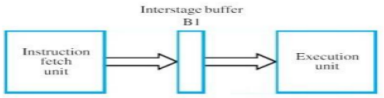
- **Instruction Decoder & Control Unit** is responsible for
  → Decoding the instruction and issuing the control-signals to all the units inside theprocessor.
  → implementing the actions specified by the instruction (loaded in the IR).
- **Processor Registers** - Register R0 through R(n-1) are also called as General Purpose Register.
  The programmer can access these registers for general-purpose use.
- **Temporary Registers** – There are 3 temporary registers in the processor. Registers
  - **Y**, **Z** & **Temp** are used for temporary storage during program-execution. The programmer cannot access these 3 registers.
- In **ALU**,1) "A" input gets the operand from the output of the multiplexer(MUX).
  2) "B" input gets the operand directly from the processor-bus.
- There are 2 options provided for "A" input of the ALU.
- MUX is used to select one of the 2 inputs.
- **MUX** selects either
  → output of Y or
  → constant-value 4(which is used to increment PC content).
- An instruction is executed by performing one or more of the following operations:



9b Describe the basic idea of instruction pipeline. [10] 5 L2

The **speed of execution of programs is influenced by many factors**.

- **One way to improve** performance is to use **faster circuit technology to build the processor and the main memory.** Another possibility is to **arrange the hardware so that more than one operation can be performed at the same time.** In this way, **the number of operations performed per second is increased even though the elapsed time needed to perform any one operation is not changed.**
- **Pipelining is a particularly effective way of organizing concurrent activity in a computer system.**
- The technique of **decomposing** a sequential process into sub-operations, with each sub-operation being executed in a dedicated segment .
- **pipelining** is commonly known as an **assembly-line** operation.

10a	Consider **how the idea of pipelining can be used in a computer**. The **processor executes a program by fetching and executing instructions, one after the other**. Let **Fi and Ei refer to the fetch and execute steps for instruction Ii** . Execution of a program consists of a sequence of fetch and execute steps, as shown in Figure a.  (a) Sequential execution  (b) Hardware organization Now consider a computer that has **two separate hardware units**, **one** for **fetching instructions** and **another for executing** them, as shown in Figure b. The instruction fetched by the fetch unit is deposited in an **intermediate storage buffer, B1**. This **buffer is needed to enable the execution unit** to execute the instruction while the fetch unit is fetching the next instruction. The results of execution are deposited in the destination location specified by the instruction.	[10]	5	L4

10	a.	Explain the process of fetching word from memory in processor.	10	L4	CO5

- To fetch instruction/data from memory, the processor has to specify the address of the memory location where this information is stored and request a Read operation.
- processor transfers required address to MAR. At the same time, processor issues Read signal on control-lines of memory-bus.
- When requested-data are received from memory, they are stored in MDR. From MDR, they are transferred to other registers in the processor.
  The Connections for register MDR has shown in fig 7.4

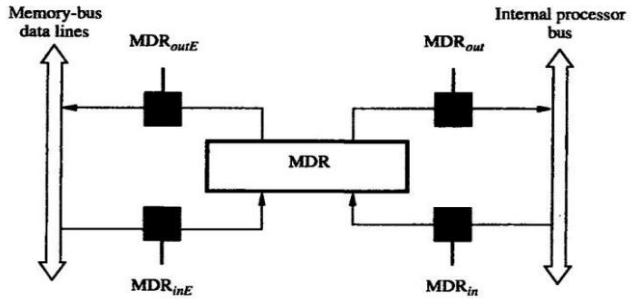


Figure 7.4 Connection and control signals for register MDR.

CONTROL-SIGNALS OF MDR

- The **MDR register has 4 control-signals** (Figure 7.4):

1) **MDRin & MDRout** control the connection to the **internal** processor data bus &
2) **MDRinE & MDRoutE** control the connection to the **external** memory Data bus.

- **Similarly, MAR register has 2 control-signals.**

The response time of each memory access varies. To accommodate this MFC is used(MFC= Memory Function Completed)

MFC=1 indicate that contents of specified location have been read and are available on the data lines of the memory bus.

- Consider **the instruction Move (R1),R2**. The action needed to execute this instruction are

1. MAR ← [R1]
2. Start a Read operation on the memory bus
3. Wait for the MFC response from the memory
4. Load MDR from the memory bus
5. R2 ← [MDR]

10b	Explain the pipeline performance of a processor and pipeline stalls.	[10]	5	L2

Pipeline Performance:

- The potential increase in performance resulting from pipelining is proportional to **the number of pipeline stages**.
- However, this increase would be achieved only if pipelined operation as depicted in Figure a could be sustained without interruption throughout program execution.
- Unfortunately, this is not the True.
- Floating point may involve many clock cycle.
- For a variety of reasons, one of the pipeline stages may not be able to complete its processing task for a given instruction in the time allotted. For example, stage E in the four stage pipeline of Figure b is responsible for arithmetic and logic operations, and one clock cycle is assigned for this task. Although this may be sufficient for most operations, some operations, such as divide, may require more time to complete. Figure shows an example in which the operation specified in instruction I2 requires three cycles to complete, from cycle 4 through cycle 6. Thus, in cycles 5 and 6, the Write stage must be told to do nothing, because it has no data to work with. Meanwhile, the information in buffer B2 must remain intact until the Execute stage has completed its operation. This means that stage 2 and, in turn, stage 1 are blocked from accepting new instructions because the information in B1 cannot be overwritten. Thus, steps D4 and F5 must be postponed as shown.

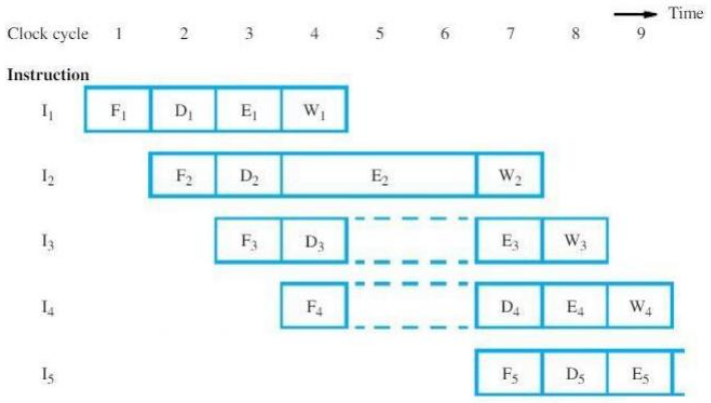


Figure 8.3 Effect of an execution operation taking more than one clock cycle. **Eg: for Data Hazard**

Pipelined operation in Figure 8.3 is said to have been **stalled for two clock cycles**. Normal pipelined operation resumes in cycle 7. Any condition that causes the pipeline to stall is called **a hazard**. We have just seen an example of a data hazard.

1) **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. As a result some operation has to be delayed, and the pipeline stalls.**

2) **control hazards** or **instruction hazards**: The pipeline may also be stalled because of a **delay in the availability of an instruction**.

For example, this may be a **result of a miss in the cache** .

3) A **third type of hazard** known as a **structural hazard**: **This is the situation when two instructions require the use of a given hardware resource at the same time.**