PVP 23

Code: 23AM3403, 23DS3403

II B.Tech - II Semester – Regular / Supplementary Examinations
APRIL 2026

DIGITAL LOGIC AND COMPUTER ORGANIZATION
(Common for AIML, DS)

Duration: 3 hours Max. Marks: 70

Note: 1. This question paper contains two Parts A and B.
2. Part-A contains 10 short answer questions. Each Question carries 2 Marks.
3. Part-B contains 5 essay questions with an internal choice from each unit. Each Question carries 10 marks.
4. All parts of Question paper must be answered in one place.

BL – Blooms Level CO – Course Outcome

PART – A

		BL	CO
1.a)	Convert $(101.01)_2$ to decimal number.	L2	CO1
1.b)	Describe the XOR gate with truth table and diagram.	L2	CO1
1.c)	What is the difference between an encoder and a decoder?	L2	CO1
1.d)	List out different types of flip-flops.	L2	CO1
1.e)	Define Indirect addressing mode.	L2	CO1
1.f)	Perform signed binary subtraction using 2's complement: 25-43	L2	CO1
1.g)	Define the terms cache hit, cache miss, cache ratio.	L2	CO1
1.h)	Differentiate between RAM and ROM in memory organization.	L2	CO1
1.i)	List out peripheral devices.	L2	CO1
1.j)	Explain how priority interrupts reduce CPU idle time.	L2	CO1

PART – B

			BL	CO	Max. Marks
UNIT-I					
2	a)	Illustrate the function $F = A + B'C$ in Canonical Sum of Product (SOP) form.	L3	CO2	5 M
	b)	Illustrate BCD and Excess 3 codes with examples.	L3	CO2	5 M
OR					
3	a)	Convert $(125.25)_{10}$ into Binary and Octal number systems.	L3	CO2	5 M
	b)	Illustrate the logic gates in memory with suitable truth tables and circuits.	L3	CO2	5 M
UNIT-II					
4	a)	Analyze the different types of multiplexers with appropriate diagrams.	L4	CO4	5 M
	b)	Illustrate Half adder, Binary adder with a truth table and logic gates.	L3	CO2	5 M
OR					
5	a)	Illustrate the operation of a master slave JK flip flops with example.	L3	CO2	5 M
	b)	Analyze Universal Shift Register.	L4	CO4	5 M
UNIT-III					
6	a)	Illustrate different types of instruction formats with examples.	L3	CO3	5 M
	b)	Describe various types of addressing modes with examples.	L2	CO1	5 M

		OR			
7	a)	Explain about Stack Organization with examples.	L2	CO1	5 M
	b)	Describe about General Register Organization with a neat diagram.	L2	CO1	5 M
		UNIT-IV			
8	a)	Analyze various types of auxiliary memories.	L4	CO4	5 M
	b)	Analyze the principle of locality of reference in cache memory.	L4	CO4	5 M
		OR			
9		Analyze the a) Associative mapping b) Direct mapping c) Set associative mapping in cache memory with relevant examples.	L4	CO4	10 M
		UNIT-V			
10	a)	Describe about i) Polling ii) Daisy Chain Priority iii) Parallel Priority Interrupt with examples	L2	CO1	5 M
	b)	Apply the role of DMA in improving system performance compared to programmed I/O.	L3	CO3	5 M
		OR			

11	a)	Illustrate the need for asynchronous data transfer in input/output organization with an example.	L3	CO3	5 M
	b)	Discuss the concept of Handshaking in detail.	L2	CO1	5 M

PVP23

Code No:23AM3403,23DS3403

II B. Tech – II Semester-Regular Examinations-April,2026
DIGITAL LOGIC AND COMPUTER ORGANIZATION
(Common to AIML, DS)

Duration: 3 Hours **Max. Marks: 70**

Note:
1.This question paper contains two Parts A and B.
2. Part-A contains 10 short answer questions. Each Question carries 2 Marks.
3. Part-B contains 5 essay questions with an internal choice from each unit. Each Question carries 10 marks.
4. All parts of Question paper must be answered in one place

PART-A

10X2=20M

Q No	Question	Marks Awarded
1(a)	**Convert $(101.01)_2$ to Decimal Number** Ans: Procedure and Answer – 2X1=2M	2M
1(b)	**Describe XOR gate with with truth table and diagram** Ans: truth table and diagram- 2X1=2M	2M
1(c)	**What is the difference between an encoder and a decoder** Ans: Any one/ two differences – 2M	2M
1(d)	**List out different types of flipflops** Ans: Any two flip flops -2X1=2M	2M
1(e)	**Define Indirect addressing mode** Ans: Definition -2M	2M
1(f)	**Perform signed binary subtraction using 2's complement: 25-43** Ans: Conversion to binary systems-1 Mark, Subtraction using 2's complement procedure-1 Mark	2M
1(g)	**Define the terms cache hit, cache miss and cache ratio** Ans: Definition of Cache hit/cache miss- 1 Mark, Cache ratio-1 Mark	2M
1(h)	**Differentiate between RAM and ROM in Memory organization** Ans: Any two differences- 2X1=2M	2M
1(i)	**List out peripheral devices** Ans: Names of any two devices -2X1=2M	2M
1(j)	**Explain how priority interrupts reduce CPU time** Ans: Any two key points relevant to topic - 2X1=2M	2M

21

PART - B

5X10=50 M

Q No	Question	Marks Awarded	Total Marks
	UNIT – I		
2(a)	**Illustrate the function F=A+B'C in canonical Sum of Product (SOP) form**		5M
	Construction of truth/function table	2 Marks	
	Determining the solution using K Map (three variables)	3 Marks	
2(b)	**Illustrate BCD and Excess 3 codes with examples**		5 M
	Explanation about BCD and Excess 3 Codes	3 Marks	
	Examples	2 Marks	
	OR		
3(a)	**Convert $(125.25)_{10}$ into Binary and Octal Number Systems**		5M
	Binary Conversion for Integer and Fraction Part	3 Marks	
	Octal Conversion for Integer and Fraction Part	2 Marks	
3(b)	**Illustrate the logic gates in memory with suitable truth tables and diagrams**		5M
	Truth tables for And, Or, Not Gates	3 Marks	
	Logic Circuits/Diagrams	2 Marks	
	UNIT-II		
4(a)	**Analyze different types of Multiplexers with appropriate diagrams**		5M
	Analysis of Two to one line Multiplexer	2 Marks	
	Analysis of Four to one line Multiplexer	3 Marks	
4(b)	**Illustrate Half Adder and Binary Adder with truth table and logic gates**		5M
	Half Adder with truth table and diagram	2 Marks	
	Binary Adder with truth table and diagram	3 Marks	
	OR		
5(a)	**Illustrate the operation of a master slave JK Flipflops with examples**		5M
	Introduction and brief description of JK Flip flop	3 Marks	
	Master slave flipflop with examples	2 Marks	
5(b)	**Analyze Universal Shift Register**		5 M
	Introduction, Definition	2 Marks	
	Explanation with examples/Diagrams	3 Marks	
	UNIT-III		
6(a)	**Illustrate different types of Instruction formats with examples**		5M
	Explanation about any three types (Zero, One, two, Three)	3 Marks	
	Examples	2 Marks	
6(b)	**Describe various types of Addressing Modes with examples**		5M
	Introduction to Addressing Modes	1 Mark	
	Any four modes with examples	4 Marks	
	OR		
7(a)	**Explain about Stack Organization with examples**		5M
	Introduction to stacks	1 Mark	
	Explanation with examples	4 Marks	
7(b)	**Describe General Register Organization with a neat diagram**		5M
	Introduction to General Register Organization	2 Marks	
	Explanation with a diagram and example	3 Marks	

	UNIT-IV		
8(a)	**Analyze various types auxiliary memories**		5M
	Introduction to Auxiliary Memory	2 Marks	
	Analysis of features of Magnetic Tape and Magnetic Disk	3 Marks	
8(b)	**Analyze Principle of locality of reference in cache memory**		5M
	Introduction to Cache memory	2 Marks	
	Discussion on locality of reference	3 Marks	
	OR		
9(a)	**Analyze** **(a) Associative Mapping** **(b) Direct Mapping** **(c) Set Associative Mapping in cache memory with relevant examples**		10 M
	Explanation with example on Associative Mapping	3 Marks	
	Explanation with example on Direct Mapping	3 Marks	
	Explanation with example on set Associative Mapping	3 Marks	
	Conclusion	1 Mark	
	UNIT - V		
10(a)	**Describe about** **(i) Polling** **(ii) Daisy Chain Priority** **(iii) Parallel priority with examples**		5M
	Explanation on Polling, Daisy Chain Priority and Parallel Priority	3 Marks	
	Examples	2 Marks	
10(b)	**Apply the role of DMA in improving system performance compared to Programmed I/O**		5M
	Introduction to DMA	2 Marks	
	Explanation on role of DMA in improving system performance	4 Marks	
	OR		
11(a)	**Illustrate the need for asynchronous data transfer in I/O organization with an example**		5M
	Introduction to Asynchronous Data transfer	2 Marks	
	Explanation with need of Asynchronous Data transfer with examples	3 Marks	
11(b)	**Discuss the concept of Handshaking in detail**		5M
	Introduction to Handshaking	1 Mark	
	Explanation with Handshaking with two types with neat diagrams/examples	4 Marks	

Code No:23AM3403,23DS3403

PVP23

II B. Tech – II Semester-Regular Examinations-April,2026
DIGITAL LOGIC AND COMPUTER ORGANIZATION
(Common to AIML, DS)

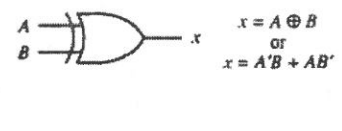
Duration: 3 Hours **Max. Marks: 70**

Note:

1.This question paper contains two Parts A and B.
2. Part-A contains 10 short answer questions. Each Question carries 2 Marks.
3. Part-B contains 5 essay questions with an internal choice from each unit. Each Question carries 10 marks.
4. All parts of Question paper must be answered in one place

PART-A

10X2=20M

Q No	Question	Marks Awarded
1(a)	**Convert $(101.01)_2$ to Decimal Number** $(101.01)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$ $1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 4 + 0 + 1 + 0 + 0.25 = 5.25$ Final Answer: 5.25_{10}	2M
1(b)	**Describe XOR gate with truth table and diagram** An **XOR (Exclusive OR) gate** gives output **1 only when the inputs are different**, and **0 when inputs are the same**. Exclusive-OR (XOR)  $x = A \oplus B$ or $x = A'B + AB'$ A B | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0	2M
1(c)	**What is the difference between an encoder and a decoder** **Decoder:** Decoder is a combinational circuit. It has N inputs and 2^N outputs. The decoder is called n-to-m-line decoder, where $m \leq 2^n$ **Encoder:** Encoders is a combinational circuit which takes 2^N inputs and gives out N outputs.	2M
1(d)	**List out different types of flipflops** 1. SR Flipflop 2. JK Flipflop 3. D Flipflop 4. Toggle Flipflop	2M
1(e)	**Define Indirect addressing mode** In Indirect addressing mode, the instruction contains the address of a memory location that holds the actual effective address (not the data itself). The CPU must perform two memory accesses — first to get the address, then to get the data.	2M

1(f)	**Perform signed binary subtraction using 2's complement: 25-43** $25 = 00011001,\ 43 = 00101011$ 2's comp(43) = 11010101 $00011001 + 11010101 = 11101110$ **(negative)** Magnitude = $00010010 = 18$ **Final Answer:** -18	2M
1(g)	**Define the terms cache hit, cache miss and cache ratio** • **Cache Hit:** When the required data is **found in cache memory**. • **Cache Miss:** When the required data is **not found in cache** and must be fetched from main memory. • **Cache Ratio (Hit Ratio):** The **ratio of cache hits to total memory accesses**.	2M
1(h)	**Differentiate between RAM and ROM in Memory organization** **RAM (Random Access Memory)** is a **volatile memory** used for temporary storage during program execution, allowing both read and write operations. **ROM (Read Only Memory)** is a **non-volatile memory** used to store permanent instructions like firmware, and it is mostly read-only.	2M
1(i)	**List out peripheral devices** **Peripheral devices** are external devices connected to a computer for input, output, or storage functions. **(Optional Part)** **Examples:** Keyboard, Mouse, Monitor, Printer, Scanner, Speakers, Hard Disk, USB drive.	2M
1(j)	**Explain how priority interrupts reduce CPU time** **Priority interrupts** allow the CPU to handle **high-priority tasks first**, instead of servicing all requests equally. This reduces CPU idle/wait time and avoids unnecessary processing of low-priority tasks, thereby **improving overall efficiency and response time**.	2M

PART - B 5X10=50 M

UNIT – I

2(a). Illustrate the function F=A+B'C in canonical Sum of Product (SOP) form 5M

Ans:

A problem could be solved by using truth table construction/Direct expansion/algebraic approach

Truth table construction:

Truth table construction --→ Rows where F=1-→Canonical SOP Form

A	B	C	B′	B′C	F = A + B′C
0	0	0	1	0	0
0	0	1	1	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	1	1	1
1	1	0	0	0	1
1	1	1	0	0	1

Minterms (Rows where F = 1)

- (A=0, B=0, C=1) → **m1**
- (A=1, B=0, C=0) → **m4**
- (A=1, B=0, C=1) → **m5**
- (A=1, B=1, C=0) → **m6**
- (A=1, B=1, C=1) → **m7**

Canonical SOP Form

F(A,B,C) =Σm(1,4,5,6,7)

Expanded:

F=A′B′C+AB′C′+AB′C+ABC′+ABC

2(b) Illustrate BCD and Excess 3 codes with examples **5M**

Ans:

BCD:

The full form of BCD is 'Binary-Coded Decimal'. Since this is a coding scheme relating decimal and binary numbers, *four bits are required* to code each decimal number.

A decimal number in BCD (8421) is the same as its equivalent binary number only when the number is between 0 and 9. A BCD number greater than 10 looks different from its equivalent binary number

Example:$(185)_{10}$= (0001 1000 0101) $_{BCD}$

Excess 3:

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

Solution.

The decimal number is	3	6	7
Add 3 to each bit	+3	+3	+3
Sum	6	9	10

Converting the above sum into 4-bit binary equivalent, we have a

4-bit binary equivalent of 0110 1001 1010

Hence, the Excess-3 code for $(367)_{10}$ = 0110 1001 1010

(OR)

3(a) Convert $(125.25)_{10}$ into Binary and Octal Number Systems **5M**

Ans:

Decimal → Binary

Integer: $125_{10} = 1111101_2$

Fraction: $0.25 \times 2 = 0.5 \rightarrow 0$

$0.5 \times 2 = 1.0 \rightarrow 1 \Rightarrow 0.01_2$

Binary: $(125.25)_{10} = (1111101.01)_2$

Binary → Octal

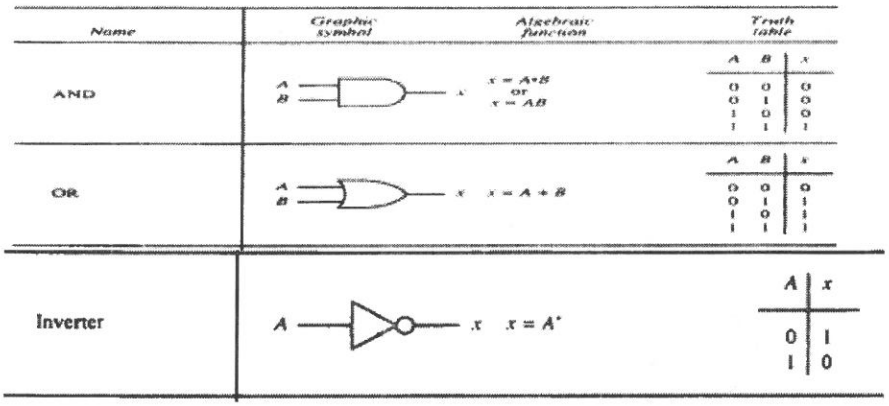
Group into 3 bits:

1111101.01 → 001 111 101.010

$(001 = 1,\ 111 = 7,\ 101 = 5,\ .010 = 2)$

Octal: $(175.2)_8$

3(b) Illustrate the logic gates in memory with suitable truth tables and diagrams 5M



Note: Inverter/Not gates are same

UNIT-II

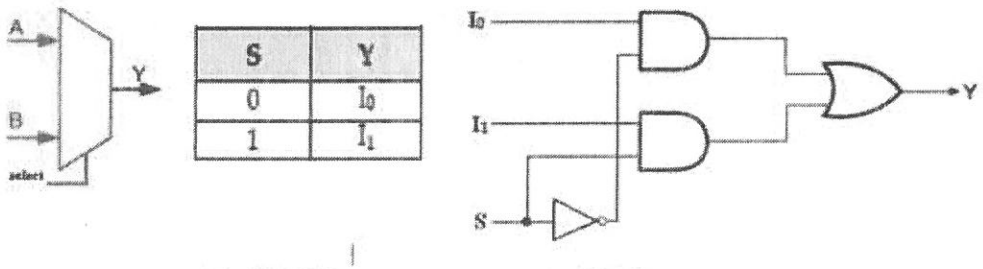
4(a) Analyze different types of Multiplexers with appropriate diagrams 5M

Ans:

- A multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. The selection of a particular input line is controlled by a set of selection lines.
- Normally, there are 2^n input lines and n selection lines whose bit combinations determine which input is selected.

2 to 1 MUX:

A 2 to 1 line multiplexer is shown in figure below, each 2 input lines A to B is applied to one input of an AND gate. Selection lines S are decoded to select a particular AND gate.

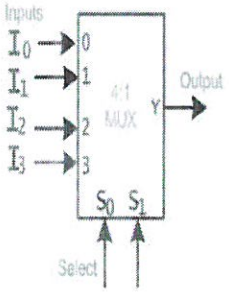


Truth table **Logic Diagram**

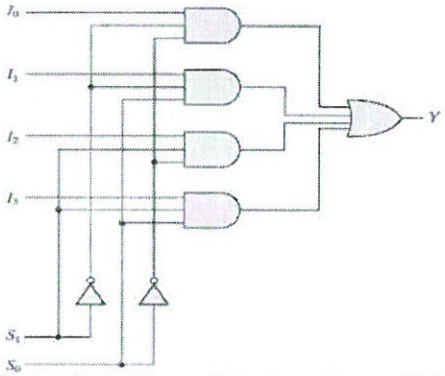
4 TO 1 MUX

- A 4 to 1 line multiplexer is shown in figure below, each of 4 inputs lines I0 to I3 is applied to one input of an AND gate.
- Selection lines S0 and S1 are decoded to select a particular AND gate.

The truth table for the 4:1 mux is given in the table below



SELECT INPUT		OUTPUT
S1	S0	Y
0	0	I0
0	1	I1
1	0	I2
1	1	I3



4(b) Illustrate Half Adder and Binary Adder with truth table and logic gates 5M

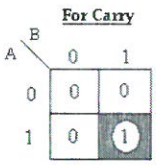
Ans:

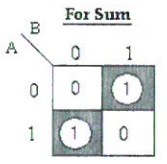
Half-Adder:

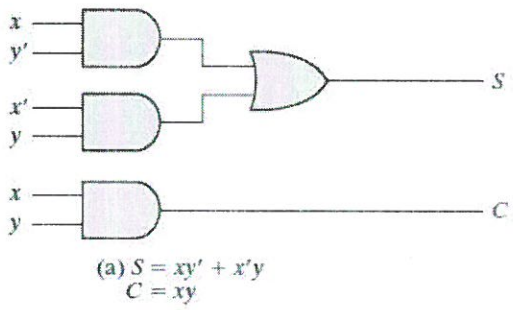
A half-adder is a combinational circuit that can be used to add two binary bits. It has **two inputs** that represent the two bits to be added and **two outputs**, with one producing the **SUM** output and the other producing the **CARRY**.

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

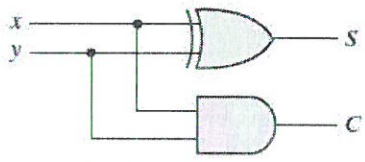
Truth table of half-adder







(a) $S = xy' + x'y$
$C = xy$



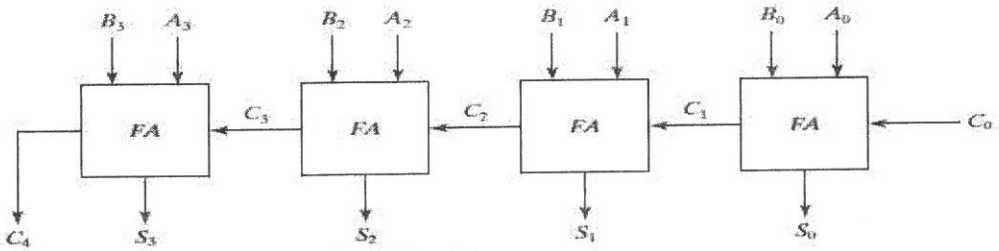
(b) $S = x \oplus y$
$C = xy$

FIGURE 4.5
Implementation of half adder

Binary Adder

- A binary adder is a digital circuit that produces the **arithmetic sum of two binary numbers.**
- It can be constructed **with full adders** connected in cascade, with the output carry from each full adder connected to the input carry of the next full adder in the chain.
- An n -bit adder requires n full adders, with each output carry connected to the input carry of the next higher order full adder.
- Consider the two binary numbers A = 1011 and B = 0011. Their sum S = 1110 is formed with the Four-bit adder as follows:

Subscript i:	3	2	1	0	
Input carry	0	1	1	0	C_i
Augend	1	0	1	1	A_i
Addend	0	0	1	1	B_i
Sum	1	1	1	0	S_i
Output carry	0	0	1	1	C_{i+1}



Four Bit Binary Adder

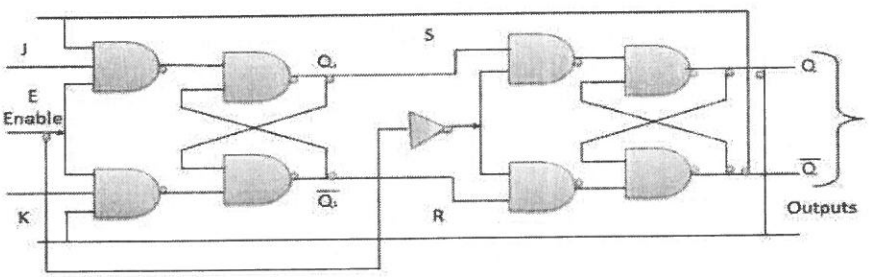
(OR)

5(a) Illustrate the operation of a master slave JK Flipflops with examples **5M**

Master Slave JK Flip Flop

Master slave JK FF is a cascade of two S-R FF with feedback from the output of second to input of first. Master is a positive level triggered. But due to the presence of the inverter in the clock line, the slave will respond to the negative level. Hence when the clock = 1 (positive level) the master is active and the slave is inactive. Whereas when clock = 0 (low level) the slave is active and master is inactive

Circuit Diagram



Truth Table

Inputs			Outputs		Comments
E	J	K	Q_{n+1}	$\overline{Q}_{n+1}$	
1	0	0	Q_n	$\overline{Q}_n$	No change
1	0	1	0	1	Rset
1	1	0	1	0	Set
1	1	1	$\overline{Q}_n$	Q_n	Toggle

5(b) Analyze Universal Shift Register **5 M**

Ans:

Universal Shift Register:

A register is capable of shifting in one direction only is a unidirectional shift register. One that can shift both directions is a bidirectional shift register. If the register has both shifts and parallel load capabilities, it is referred to as a universal shift register. Universal shift register is a bidirectional register, whose input can be either in serial form or in parallel form and whose output also can be in serial form or I parallel form.

The most general shift register has the following capabilities.

- A shift-left control to enable the shift-left operation and serial input and output lines associated with the shift-left.
- A parallel loads control to enable a parallel transfer and the n input lines associated with the parallel transfer N parallel output lines
- A control state that leaves the information in the register unchanged in the presence of the clock.

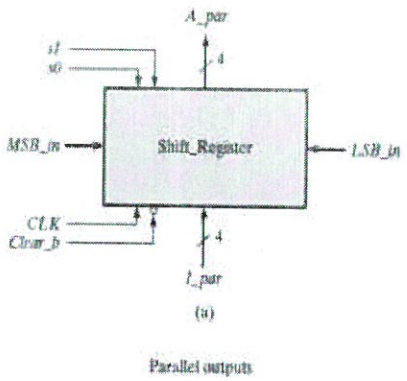


Table 6.3
Function Table for the Register of Fig. 6.7

Mode Control		
s_1	s_0	Register Operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

UNIT-III

6(a) Illustrate different types of Instruction formats with examples **5M**

Ans:

Instruction formats define how an instruction is structured in terms of opcode and operands. Based on the number of address fields, instruction formats are classified as follows:

1. **Three-Address Instruction Format**
   - Contains three operands: two source operands and one destination
2. **Two-Address Instruction Format**
   - Contains two operands
   - One operand acts as both source and destination
3. **One-Address Instruction Format**
   - Uses one explicit operand
   - Other operand is implied (usually Accumulator (AC))
4. **Zero-Address Instruction Format**
   - No explicit operands
   - Operates on stack

Format	No. of Operands	Example Instruction	Key Idea
Three-Address	3	ADD R1, R2, R3	R1 = R2 + R3
Two-Address	2	ADD R1, R2	R1 = R1 + R2
One-Address	1	ADD X	AC = AC + X
Zero-Address	0	ADD	Stack operation

6(b) Describe various types of Addressing Modes with examples **5M**

Ans:

Addressing modes define how the CPU locates the operand (data) required for executing an instruction. Different modes improve flexibility, efficiency, and speed of execution.

Mode	Example	Description
Immediate	MOV R1, #50	Constant value
Direct	MOV R1, 2000	Direct memory address
Indirect	MOV R1, (2000)	Address of address
Register	ADD R1, R2	Operand in register
Register Indirect	MOV R1, (R2)	Register holds address
Indexed	MOV R1, 1000(R2)	Base + index
Base Register	MOV R1, 20(R3)	Base + displacement
Relative	JMP 40	PC relative
Auto Increment	MOV R1, (R2)+	Auto update
Implied	CMA	Operand implied

Example

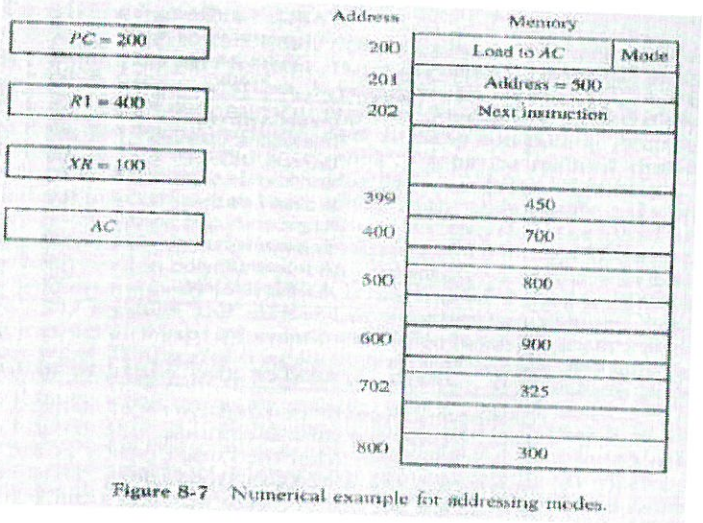


Figure 8-7 Numerical example for addressing modes.

TABLE 8-4 Tabular List of Numerical Example

Addressing Mode	Effective Address	Content of *AC*
Direct address	500	800
Immediate operand	201	500
Indirect address	800	300
Relative address	702	325
Indexed address	600	900
Register	—	400
Register indirect	400	700
Autoincrement	400	700
Autodecrement	399	450

(OR)

7(a) Explain about Stack Organization with examples **5M**

Ans:

A stack or last-in first-out (LIFO) is useful feature that is included in the CPU of most computers.

Stack:

- A stack is a storage device that stores information in such a manner that the item stored last is the first item retrieved.
- The operation of a stack can be compared to a stack of trays. The last tray placed on top of the stack is the first to be taken off.
- In the computer stack is a memory unit with an address register that can count the address only.
- The register that holds the address for the stack is called a stack pointer (SP). It always points at the top item in the stack.
- The two operations that are performed on stack are the insertion and deletion.
- The operation of insertion is called PUSH.
- The operation of deletion is called POP.
- These operations are simulated by incrementing and decrementing the stack pointer register (SP).

Types:

Register Stack and Memory Stack

Examples; Any relevant examples

7(b) Describe General Register Organization with a neat diagram **5M**

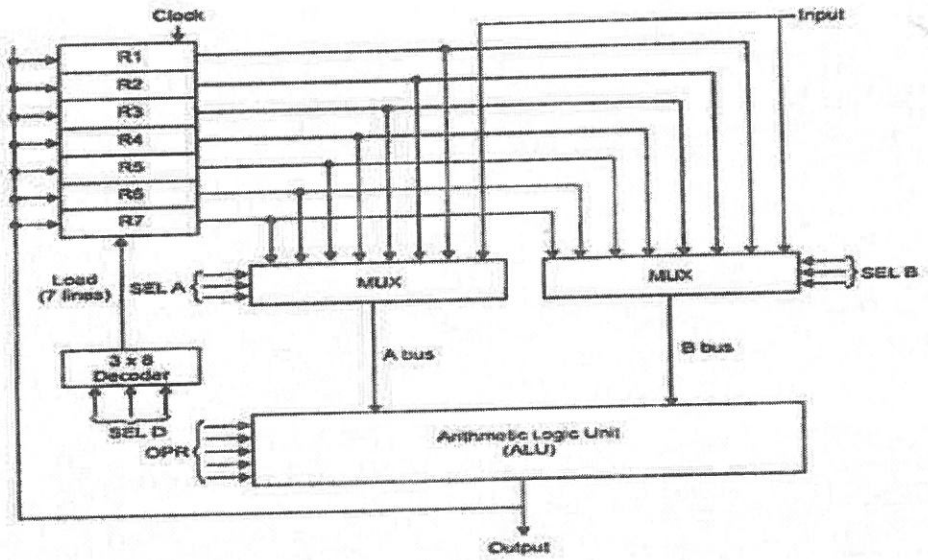
General Register Organization refers to the internal arrangement of multiple registers inside the CPU that are used for storing data, instructions, and intermediate results during program execution. It enables fast data processing by reducing memory access

A CPU contains a set of general-purpose registers (R0, R1, R2, …, Rn)

These registers are interconnected through:

- Multiplexers (MUX)
- Common data bus
- Arithmetic Logic Unit (ALU)

All operations are performed on data stored in registers.



EXAMPLE:

Example: R1 = *R2 + R3*

To perform the above operation, the control must provide binary selection variables to the following selector inputs:

1. MUX A selector (SELA): to place the content of R2 into bus A.
2. MUX B selector (SELB): to place the content of R3 into bus B.
3. ALU operation selector (OPR): to provide the arithmetic addition A + B.
4. Decoder destination selector (SELD): to transfer the content of the output bus into R1
5. MUX A selector (SELA): to place the content of R2 into bus A.
6. MUX B selector (SELB): to place the content of R3 into bus B.
7. ALU operation selector (OPR): to provide the arithmetic addition A + B.
8. Decoder destination selector (SELD): to transfer the content of the output bus into R1

UNIT-IV

8(a) Analyze various types auxiliary memories **5M**

Ans:

Auxiliary memory (also called secondary memory) is used for permanent storage of data and programs. Unlike primary memory, it is non-volatile, large in capacity, and comparatively slower. It supports long-term data retention and backup.

Types of Auxiliary Memories

Analysis Table

Type	Access Type	Speed	Cost	Usage
Magnetic Disk	**Direct**	**Medium**	**Moderate**	**General storage**
Magnetic Tape	**Sequential**	**Slow**	**Low**	**Backup/archival**
Optical Memory	**Direct**	**Slow**	**Low**	**Media storage**
Solid-State	**Direct**	**Very Fast**	**High**	**Modern systems**
Cloud Storage	**Network-based**	**Depends**	**Flexible**	**Remote storage**

Advantages of Auxiliary Memory

- Non-volatile (permanent storage)
- Large storage capacity
- Cost-effective for bulk data

Disadvantages

- Slower than primary memory
- Some types (like tape) have limited access speed
- May require additional hardware or internet

8(b) Analyze Principle of locality of reference in cache memory **5M**

Ans:

The principle of locality of reference explains why cache memory is effective in improving system performance. It states that a program tends to access a small portion of its address space repeatedly within a short period of time.

Type of Locality	Definition	Example	Cache Implication
Temporal Locality	Recently accessed data is likely to be accessed again soon	Reusing variable in a loop (`sum = sum + A`)	Keep recently used data in cache
Spatial Locality	Nearby memory locations are likely to be accessed	Accessing array elements (`arr[i]`)	Fetch block of data into cache
Sequential Locality	Instructions are executed in sequential order	Instruction1 → Instruction2 → Instruction3	Prefetch next instructions

(OR)

9(a) Analyze

(a) Associative Mapping
(b) Direct Mapping
(c) Set Associative Mapping in cache memory with relevant examples **10 M**

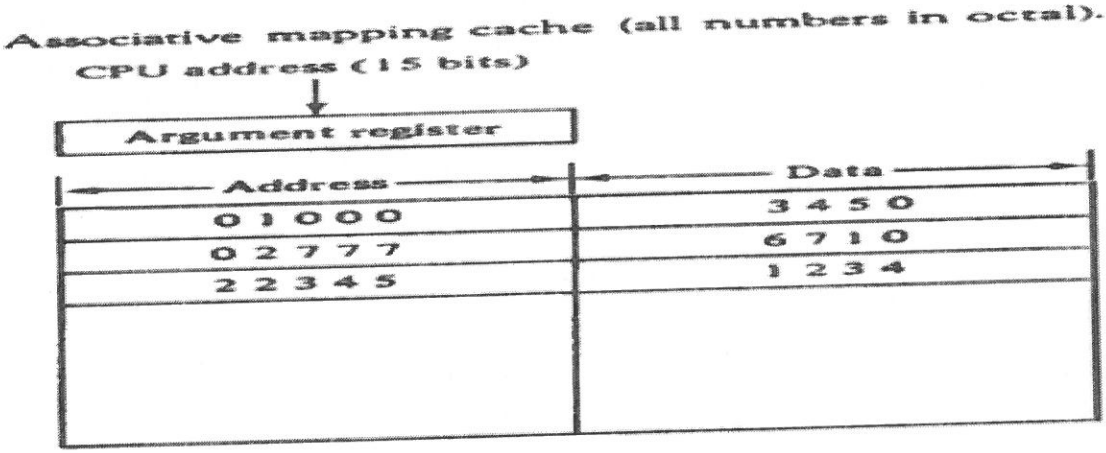
Ans:

Cache mapping techniques define how blocks of main memory are placed into cache memory. Since cache is smaller than main memory, an efficient mapping method is required to decide where a particular memory block should be stored in the cache.
The three main mapping techniques are:

- Associative Mapping → Any block can go anywhere
- Direct Mapping → Each block has a fixed location
- Set-Associative Mapping → Block maps to a set, but flexible within the set

Associative Mapping

The faster and most flexible cache organization use an associative memory. The associative memory stores both the address and content (data) of the memory word. This permits any location in cache to store any word from main memory



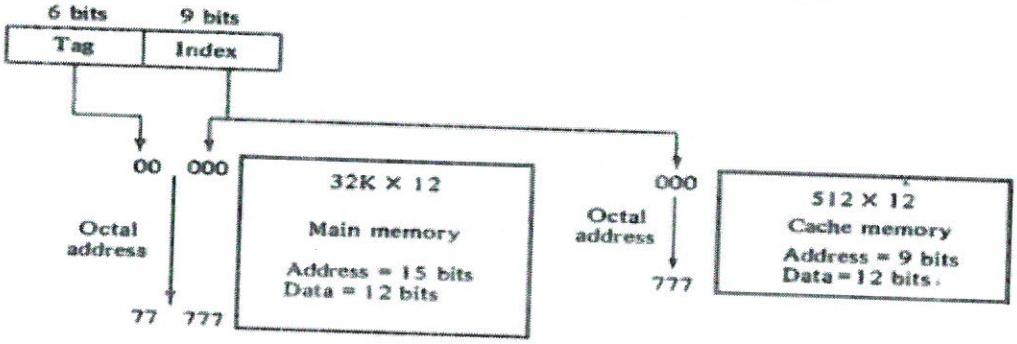
- A CPU address of 15 bits is placed in the argument register and the associative memory is searched for a matching address. If the address is found, the corresponding 12-bit data is read and sent to the CPU.
- If no match occurs, the main memory is accessed for the word. The address-data pair is then transferred to the associative cache memory. If the cache is full, an address–data pair must be displaced to make room for a pair that is needed and not presently in the cache.

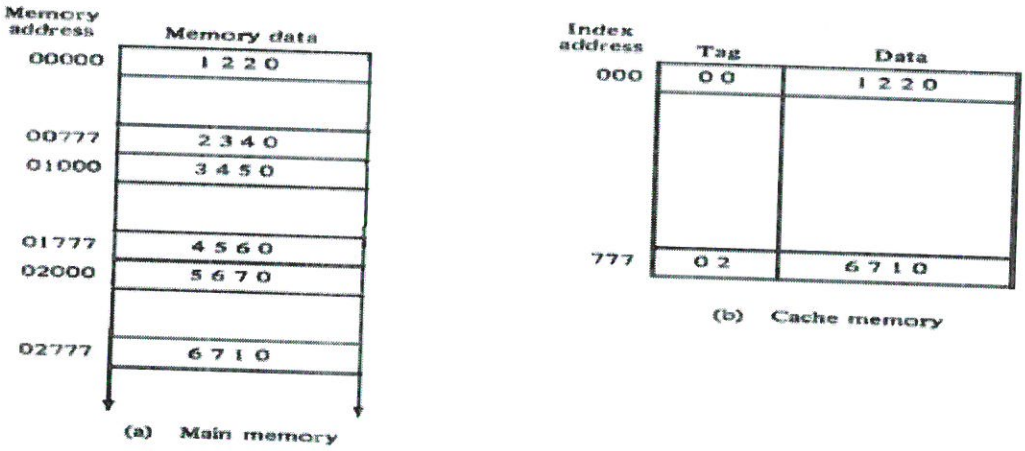
Direct Mapping

- Associative memories are expensive compared to random-access memories because of the added logic associated with each cell.
- Direct mapping uses RAM instead of CAM.

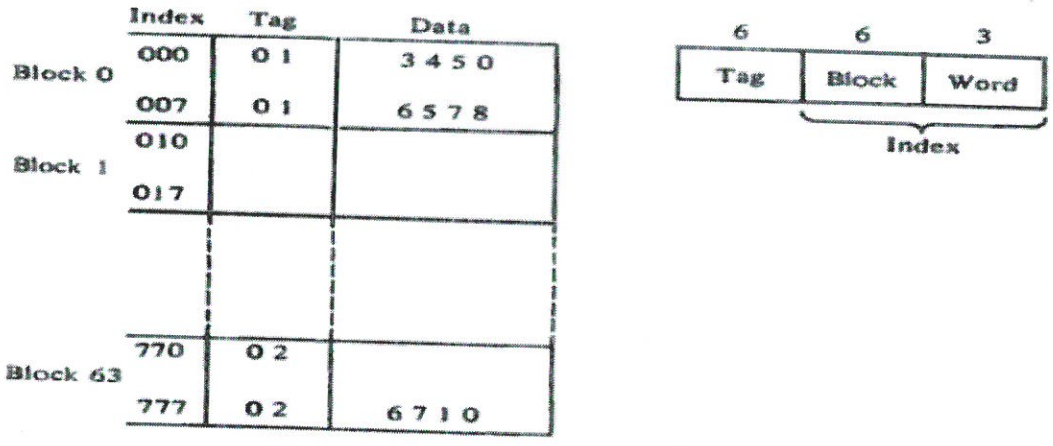
The n-bit memory address is divided into two fields: k bits for the index field and n-k bits for the tag field. The direct mapping cache organization uses the n-bit address to access the main memory and the k-bit index to access the cache

Addressing relationships between main and cache memories.





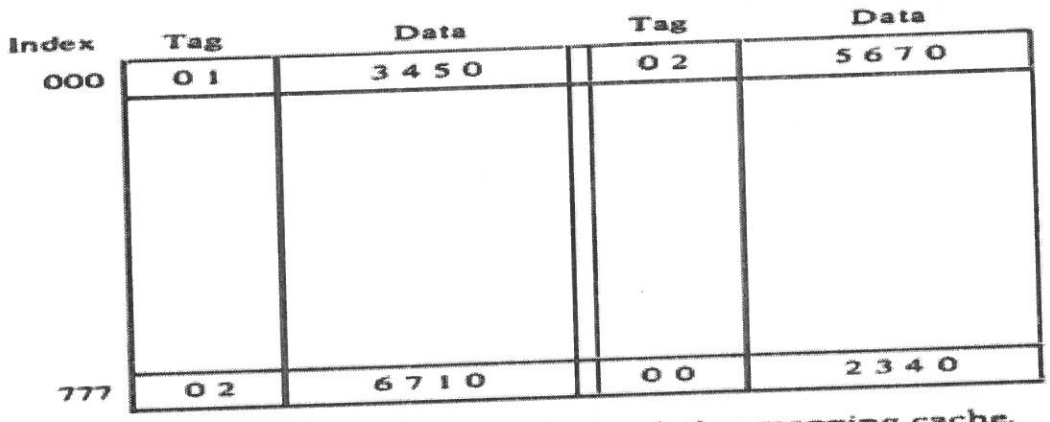
Direct mapping cache organization.



Direct mapping cache with block size of 8 words.

Set Associative Mapping

- Set-associative mapping is an improvement over the direct-mapping organization in that each word of cache can store two or more words of memory under the same index address.
- Each data word is stored together with its tag and the number of tag-data items in one word of cache is said to form a set.
- Each index address refers to two data words and their associated tags. Each tag requires six bits and each data word has 12 bits, so the word length is $2(6 + 12) = 36$ bits. An index address of nine bits can accommodate 512 words. Thus, the size of cache memory is 512×36.
- The words stored at addresses 01000 and 02000 of main memory are stored in cache memory at index address 000. Similarly, the words at addresses 02777 and 00777 are stored in cache at index address 777.



Two-way set-associative mapping cache.

UNIT - V

10(a) Describe about

(i) Polling
(ii) Daisy Chain Priority
(iii) Parallel priority with examples **5M**

Ans:

Polling Procedure:

- There is one common branch address for all interrupts.
- Branch address contains the code that polls the interrupt sources in sequence. The highest priority is tested first.
- The particular service routine of the highest priority device is served.
- The disadvantage is that time required to poll them can exceed the time to serve them in large number of IO devices.
- **Using Hardware**: Hardware priority system function as an overall manager.
- It accepts interrupt request and determines the priorities.
- To speed up the operation each interrupting devices has its own interrupt vector. No polling is required, all decision are established by hardware priority interrupt unit.
- It can be established by serial or parallel connection of interrupt lines

Serial or Daisy Chaining Priority:

- Device with highest priority is placed first.
- Device that wants the attention send the interrupt request to the CPU.
- CPU then sends the INTACK signal which is applied to PI(priority in) of the first device.
- If it had requested the attention, it place its VAD(vector address) on the bus. And it block the signal by placing 0 in PO(priority out)
- If not it pass the signal to next device through PO(priority out) by placing 1.
- This process is continued until appropriate device is found.
- The device whose PI is 1 and PO is 0 is the device that send the interrupt request

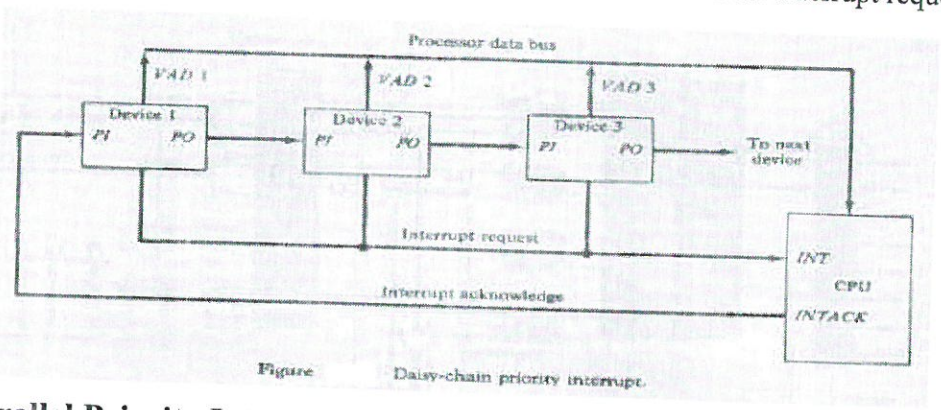


Figure Daisy-chain priority interrupt.

Parallel Priority Interrupt:

- It consists of interrupt register whose bits are set separately by the interrupting devices.
- Priority is established according to the position of the bits in the register.
- Mask register is used to provide facility for the higher priority devices to interrupt when lower priority device is being serviced or disable all lower priority devices when higher is being serviced.
- Corresponding interrupt bit and mask bit are ANDed and applied to priority encoder.
- Priority encoder generates two bits of vector address.
- Another output from it sets IST(interrupt status flip flop).

10(b) Apply the role of DMA in improving system performance compared to Programmed I/O **5M**

Ans:

Programmed I/O (PIO) requires the CPU to continuously monitor and transfer data between I/O device and memory. Direct Memory Access (DMA) allows data transfer directly between I/O device and memory without continuous CPU involvement, improving system performance.

Aspect	Programmed I/O (PIO)	DMA (Direct Memory Access)	Performance Impact
CPU Involvement	CPU handles every data transfer	CPU only initiates transfer	DMA frees CPU for other tasks
Data Transfer Speed	Slow (CPU-controlled)	Fast (direct transfer)	Higher throughput in DMA
CPU Utilization	High CPU usage (busy waiting)	Low CPU usage	Better multitasking
Parallelism	No parallel operation	CPU and I/O work simultaneously	Improves overall efficiency
Interrupts	Frequent interrupts	Minimal interrupts (only completion)	Reduces overhead
Block Transfer	Transfers one word at a time	Transfers large blocks	Faster bulk data transfer

(OR)

11(a) Illustrate the need for asynchronous data transfer in I/O organization with an example **5M**

Ans:

Introduction to Asynchronous Data transfer

Asynchronous data transfer between two independent units requires that control signals be transmitted between the communicating units to indicate the time at which data is being transmitted

In this method two types of techniques are used based on signals before data transfer:

(a) Strobe Control
(b) Handshaking

In computer systems, the **CPU and I/O devices operate at different speeds**.

- CPU → very fast
- I/O devices (keyboard, printer, disk) → comparatively slow

Because of this mismatch, **synchronous data transfer (same timing)** is inefficient. Hence, **asynchronous data transfer** is required.

What is Asynchronous Data Transfer?

- Data transfer occurs **without a common clock**
- Uses **control signals (handshaking)** to coordinate communication

Ensures data is transferred **only when both sender and receiver are ready**

Need for Asynchronous Transfer

Reason	Explanation
Speed Mismatch	CPU is faster than I/O devices
Avoids Data Loss	Ensures receiver is ready before sending data
Efficient Communication	No need to wait for fixed clock cycles
Flexibility	Works with devices of different speeds
Reliability	Uses control signals for safe transfer

Example(1): Keyboard Input

1. User presses a key
2. Keyboard sends **data + request signal**
3. CPU checks and sends **acknowledge signal**
4. Data is transferred to memory

Since keyboard is slow, asynchronous transfer ensures:

- CPU does not wait unnecessarily
- No data is lost

11(b) Discuss the concept of Handshaking in detail **5M**

The handshaking method solves the problem of strobe method by introducing a second control signal that provides a reply to the unit that initiates the transfer.

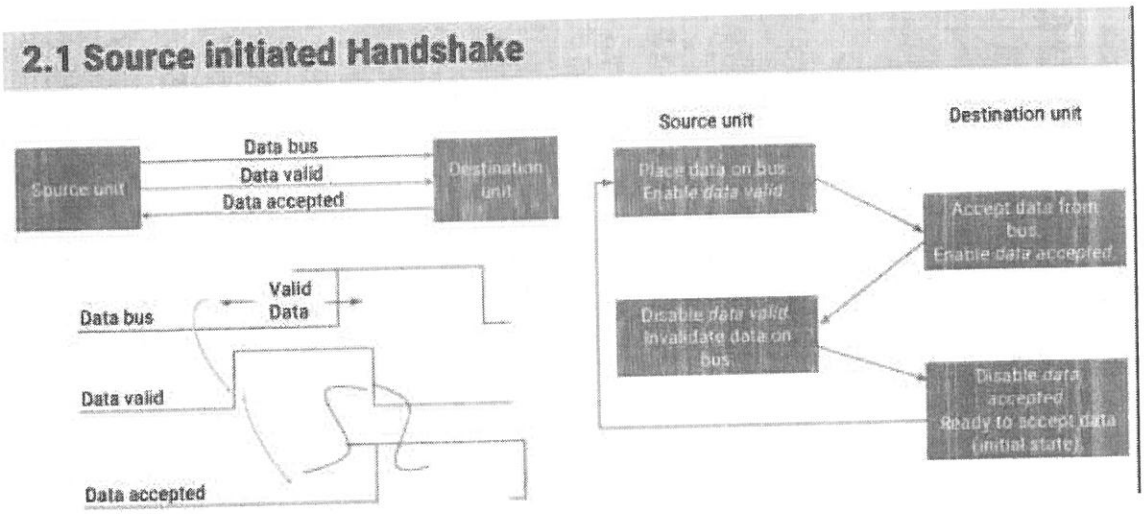
Principle of Handshaking:

The basic principle of the two-wire handshaking method of data transfer is as follow:

One control line is in the same direction as the data flows in the bus from the source to destination. It is used by source unit to inform the destination unit whether there a valid data in the bus. The other control line is in the other direction from the destination to the source. It is used by the destination unit to inform the source whether it can accept the data. The sequence of control during the transfer depends on the unit that initiates the transfer.

(a) Source Initiated Transfer using Handshaking:

The sequence of events shows **four possible states** that the system can be at any given time. The source unit **initiates the transfer** by placing the data on the bus and enabling its data valid signal. The **data accepted signal** is activated by the destination unit after it accepts the data from the bus. The source unit then **disables its data** accepted signal and the system goes into its **initial state**. The sequence of events shows four possible states that the system can be at any given time



(b) Destination Initiated Transfer Using Handshaking:

The name of the signal generated by the destination unit has been changed to ready for data to reflects its new meaning. The source unit in this case does not place data on the bus until after it receives the ready for data signal from the destination unit. From there on, the handshaking procedure follows the same pattern as in the source-initiated case. The only difference between the Source Initiated and the Destination Initiated transfer is in their choice of Initial sate