

Diploma Engineering

Laboratory Manual

Computer Organization & Architecture (4350701)

[Computer Engineering, Semester V]

Enrolment No

Name

Branch

Academic Term

Institute



Directorate of Technical Education
Gandhinagar – Gujarat

COMPUTER ORGANIZATION & ARCHITECTURE (4350701)

Laboratory Manual

Prepared By,

Shri. S. B. Prasad,
Lecturer, Computer Engineering
Govt. Polytechnic, Gandhinagar

Shri. Jiger P. Acharya
Lecturer, Computer Engineering
Govt. Polytechnic, Ahmedabad

Shri. Niraj R. Trivedi
Lecturer, Compute Engineering
A. V. Parekh Technical Institute, Rajkot

Branch Co-Ordinator,
Shri. B. H. Kantevala
HoD, Computer Engineering,
Govt. Polytechnic, Ahmedabad

Committee Chairman,
Shri. R. D. Raghani
HoD, Electronics & Communication,
Govt. Polytechnic, Gandhinagar

DTE's Mission:

- To provide globally competitive technical education;
- Remove geographical imbalances and inconsistencies;
- Develop student friendly resources with a special focus on girls' education and support to weaker sections;
- Develop programs relevant to industry and create a vibrant pool of technical professionals.

Institute's Vision:

Institute's Mission:

Department's Vision:

Department's Mission:



A. V. PAREKH TECHNICAL INSTITUTE, RAJKOT
COMPUTER ENGINEERING DEPARTMENT

Certificate

This is to certify that Mr./Ms
Enrollment No. of **5th** Semester of **Diploma in**
Computer Engineering of
(GTU Code) has satisfactorily completed the term work in course
..... for the
Academic year: Term: **Odd/Even** **prescribed** in the GTU curriculum.

Place: Rajkot

Date:

Signature of Course Faculty

Head of the Department

Preface

The primary aim of any laboratory/Practical/field work is enhancement of required skills as well as creative ability amongst students to solve real time problems by developing relevant competencies in psychomotor domain. Keeping in view, GTU has designed competency focused outcome-based curriculum - 2021 (COGC-2021) for Diploma engineering programmes. In this more time is allotted to practical work than theory. It shows importance of enhancement of skills amongst students and it pays attention to utilize every second of time allotted for practical amongst Students, Instructors and Lecturers to achieve relevant outcomes by performing rather than writing practice in study type. It is essential for effective implementation of competency focused outcome- based Green curriculum-2021. Every practical has been keenly designed to serve as a tool to develop & enhance relevant industry needed competency in each and every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual has been designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual, students can read procedure one day in advance to actual performance day of practical experiment which generates interest and also, they can have idea of judgement of magnitude prior to performance. This in turn enhances predetermined outcomes amongst students. Each and every Experiment / Practical in this manual begins by competency, industry relevant skills, course outcomes as well as practical outcomes which serve as a key role for doing the practical. The students will also have a clear idea of safety and necessary precautions to be taken while performing experiment.

This manual also provides guidelines to lecturers to facilitate student-cantered lab activities for each practical/experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve outcomes. It also gives an idea that how students will be assessed by providing Rubrics.

Students are eventually going to become a computer engineer and will be placed in an industry. If he/she only possess the knowledge about the languages and its programming then they would have been lacking in some aspects like how the memory and I/O works as well as how the CPU and its components work inside the hardware. By studying this curriculum and performing various practical / exercises given in this manual they can gain the full knowledge of each component's working and their interconnection for communication hence providing full internal knowledge.

Although we tried our level best to design this lab manual, but always there are chances of improvement. We welcome any suggestions for improvement.

Programme Outcomes (POs)

Following programme outcomes are expected to be achieved through the practical of the course:

1. **Basic and Discipline specific knowledge:** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
2. **Problem analysis:** Identify and analyse well-defined *engineering* problems using codified standard methods.
3. **Design/development of solutions:** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing:** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.
5. **Engineering practices for society, sustainability and environment:** Apply appropriate technology in context of society, sustainability, environment and ethical practices.
6. **Project Management:** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
7. **Life-long learning:** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

Practical Outcome - Course Outcome matrix

Course Outcomes (COs):						
<p>a. <u>CO1</u>: Analyse computer systems at the hardware level, including CPU components & circuits, buses, and registers considering trade-offs and the evolution of processors.</p> <p>b. <u>CO2</u>: Examine 8085 Architecture and its working.</p> <p>c. <u>CO3</u>: Perform Assembly language programming using 8085 Instruction Set.</p> <p>d. <u>CO4</u>: Characterize need of various Memory types in hierarchy.</p> <p>e. <u>CO5</u>: Visualize CPU-I/O Communication and working.</p>						
Sr. No.	Experiment/Practical Outcome	CO1	CO2	CO3	CO4	CO5
1	<u>Processor Evolution</u> Outline intel processor evolution	√	-	-	-	-
2	<u>8085 Architecture</u> Prepare 8085 Microprocessor architecture in diagram and explain it	-	√	-	-	-
3	<u>Data Transfer Instructions</u> Summarize out Data Transfer Instructions and perform minimum 3 to 5 programs associated with the said concept	-	-	√	-	-
4	<u>Arithmetic Instructions</u> Summarize Arithmetic Instructions and perform minimum 3 to 5 programs associated with the said concept	-	-	√	-	-
5	<u>Logical Instructions</u> Summarize Logical Instructions of 8085 with example and execute minimum 3 to 5 programs associated with said concept.	-	-	√	-	-
6	<u>I/O Instructions</u> List Input-Output Instructions of 8085 with example and execute minimum 3 to 5 programs associated with said concept.	-	-	√	-	-
7	<u>Machine Control Instructions</u> Recall Machine Control Instructions of 8085 with example and execute minimum 2 to 3 programs associated with said concept.	-	-	√	-	-
8	<u>Branching & Looping Instructions</u> List Branching and Looping instructions of 8085 with example and execute basic 2-3 programs associated with said concept	-	-	√	-	-
9	<u>Memory Hierarchy</u> Make a chart to represent all types of memory in Memory Hierarchy.	-	-	-	√	-
10	<u>Associative Memory</u> Paraphrase Associative Memory in details	-	-	-	√	-
11	<u>Input – Output</u> Differentiate Programmed I/O and Interrupt initiated I/O in detail	-	-	-	-	√
12	<u>CPU-IOP Communication</u> List steps to carryout CPU-IOP Communication.	-	-	-	-	√

Industry Relevant Skills

The following industry relevant skills of the competency “**Examine computer architecture and explore assembly language programming using 8085 instructions set**” are expected to be developed in the student by undertaking the practical of this laboratory manual.

1. Prepare comparison table for various processors.
2. Depict processor architecture with all its components
3. Write code in assemble language using various instructions
4. Summarize various memories and their usage and working methodology
5. Explore IOP and CPU-IOP communication.

Guidelines to Teachers

1. Couse faculty should demonstrate experiment with all necessary implementation strategies described in curriculum.
2. Couse faculty should explain industrial relevance before starting of each experiment.
3. Course faculty should involve & give opportunity to all students for hands on experience.
4. Course faculty should ensure mentioned skills are developed in the students by asking.
5. Utilise 2 hrs of lab hours effectively and ensure completion of write up with quiz also.
6. Encourage peer to peer learning by doing same experiment through fast learners.

Instructions for Students

1. Organize the work in the group and make record of all observations.
2. Students shall develop maintenance skill as expected by industries.
3. Student shall attempt to develop related hand-on skills and build confidence.
4. Student shall develop the habits of evolving more ideas, innovations, skills etc.
5. Student shall refer technical magazines and data books.
6. Student should develop habit to submit the practical on date and time.
7. Student should well prepare while submitting write-up of exercise.

RUBRICS FOR EVALUATION

▪ **CONTINUOUS ASSESSMENT (25 Marks):**

- **Laboratory Work and Questionnaire Component (25 Marks):**

Component	Criteria	Percentage	Assessment
Laboratory Work and Questionnaire	Excellent	91%-100%	Demonstrates exceptional proficiency in both laboratory work and questionnaire assessments, consistently applying skills and understanding effectively.
	Proficient	71%-90%	Shows a strong command of both laboratory work and questionnaire assessments, with minor areas for improvement.
	Satisfactory	51%-70%	Achieves a satisfactory level of performance in laboratory work and questionnaire assessments, with room for improvement in some areas.
	Needs Improvement	31%-50%	Demonstrates limited proficiency in both laboratory work and questionnaire assessments, with significant areas for improvement.
	Inadequate	0%-30%	Fails to meet acceptable standards in both laboratory work and questionnaire assessments; significant improvement is required.

TABLE OF CONTENTS

Enrollement No.: _____

Term: 2024-25 (ODD)

Name: _____

Sr. No	Experiment/Practical Outcome	Page	Date Performed	Marks (25)	Sign
1	<u>Processor Evolution</u> Outline intel processor evolution				
2	<u>8085 Architecture</u> Prepare 8085 Microprocessor architecture in diagram and explain it				
3	<u>Data Transfer Instructions</u> Summarize out Data Transfer Instructions and perform minimum 3 to 5 programs associated with the said concept				
4	<u>Arithmetic Instructions</u> Summarize Arithmetic Instructions and perform minimum 3 to 5 programs associated with the said concept				
5	<u>Logical Instructions</u> Summarize Logical Instructions of 8085 with example and execute minimum 3 to 5 programs associated with said concept.				
6	<u>I/O Instructions</u> List Input-Output Instructions of 8085 with example and execute minimum 3 to 5 programs associated with said concept.				
7	<u>Machine Control Instructions</u> Recall Machine Control Instructions of 8085 with example and execute minimum 2 to 3 programs associated with said concept.				
8	<u>Branching & Looping Instructions</u> List Branching and Looping instructions of 8085 with example and execute basic 2-3 programs associated with said concept				
9	<u>Memory Hierarchy</u> Make a chart to represent all types of memory in Memory Hierarchy.				
10	<u>Associative Memory</u> Paraphrase Associative Memory in details				
11	<u>Input – Output</u> Differentiate Programmed I/O and Interrupt initiated I/O in detail				
12	<u>CPU-IOP Communication</u> List steps to carryout CPU-IOP Communication.				

Practical No. 01: Processor Evolution

1. Outline intel processor evolution.

A. Objectives:

Basic understanding of various aspects in which processors are termed as different in terms of its architecture develops a sense of hardware and its working capability.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Engineering Tools, Experimentation and Testing (PO4):** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes *in field of engineering*.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Examine computer architecture**’:

1. Understanding Processor word-size
2. Compute Maximum Physical Memory Addressable.

D. Relevant Course Outcomes (COs):

1. Explain Generic Computer and ALU Architecture and processor Evolution.

E. Practical Outcomes:

1. Differentiate and choose processor based on the current need.

F. Relevant Affective domain Outcomes (ADOs):

1. Understand need of the scenario and develop a system accordingly.

G. Prerequisite Theory:

Microprocessor is considered as the mind of the computer system which performs all the computational task of Arithmetical, Logical or any other Nature. It is available in the CPU of the computer system.

Microprocessor has 3 buses:

1. Address Bus: It is a unidirectional bus providing addresses generated by CPU to other devices
 2. Data Bus: It is a bi-directional bus to communicate data between microprocessor and Peripheral devices.
 3. Control Bus: Control bus is actually a set of Control lines in which each line provides a specific signal like Read, Write Operation or I/O or Memory Operation.
- Word Size: The data that can be processed by the microprocessor in a single cycle is known as the word size. It usually varies from 4-bits to 64-bits depending upon the processor.
 - Addressable Physical Memory: Depending upon the size of Address Bus (number of address lines) we can identify number of unique memory locations generated by the microprocessor
 - For example: If a processor has 4 bits of Address But then number of unique addresses generated are 0000 (0d) to 1111 (15d). 16 total locations. This can be calculated using $2^{(\text{number of address lines})}$

H. Comparison:

Sr. No.	Microprocessor	Invention Year	Word Size	Size of Physical Memory	Virtual Memory Supported?
1	Intel 4004				
2	Intel 8008				
3	Intel 8085				
4	Intel 8086				
5	Intel 80186				
6	Intel 80286				
7	Intel 80386				
8	Intel 80486				
9	Intel Pentium I				
10	Intel Pentium II				
11	Intel Pentium III				
12	Intel Pentium IV				
13	Intel Core i3				
14	Intel Core i5				
15	Intel Core i7				
16	Intel Core i9				

I. Practical related Quiz:

1. What was the word size of the Intel 4004 microprocessor?
 - a) 4 bits
 - b) 8 bits
 - c) 16 bits
 - d) 32 bits
2. Which microprocessor had the largest physical memory size among the options below?
 - a) Intel 8086
 - b) Intel 80286
 - c) Intel 80386
 - d) Intel 80486
3. In which year was the Intel 8008 microprocessor introduced?
 - a) 1969
 - b) 1971
 - c) 1974
 - d) 1978
4. What was the word size of the Intel 8086 microprocessor?
 - a) 8 bits
 - b) 16 bits
 - c) 32 bits
 - d) 64 bits
5. Which microprocessor introduced the x86 architecture?
 - a) Intel 4004
 - b) Intel 8086
 - c) Intel 80286
 - d) Intel 80386
6. What was the word size of the Intel 80286 microprocessor?
 - a) 8 bits
 - b) 16 bits
 - c) 32 bits
 - d) 64 bits

7. In which year was the Intel 80386 microprocessor released?
 - a) 1978
 - b) 1981
 - c) 1985
 - d) 1989
8. Which microprocessor introduced the concept of the superscalar architecture?
 - a) Intel 80486
 - b) Intel Pentium
 - c) Intel Core 2 Duo
 - d) Intel Core i9
9. What was the word size of the Intel Pentium microprocessor?
 - a) 16 bits
 - b) 32 bits
 - c) 64 bits
 - d) 128 bits
10. In which year was the Intel Core i9 series of microprocessors first introduced?
 - a) 2006
 - b) 2009
 - c) 2011
 - d) 2017

Answers:

- a) 4 bits
- d) Intel 80486
- b) 1971
- b) 16 bits
- b) Intel 8086
- c) 32 bits
- c) 1985
- b) Intel Pentium
- b) 32 bits
- d) 2017

Signature

Practical No. 02: 8085 Architecture

1. Prepare 8085 Microprocessor architecture in diagram and explain it.

A. Objectives:

Understand the basic structure and components of the 8085 microprocessor, including its registers, instruction set, and data transfer mechanisms.

Gain hands-on experience in programming and executing instructions using the 8085-assembly language.

Learn how to design and implement simple programs using the 8085 microprocessor, including arithmetic and logical operations, data manipulation, and control flow.

Develop an understanding of the timing and control signals involved in the execution of instructions in the 8085 microprocessor.

Acquire practical knowledge of interfacing peripheral devices, such as input/output ports and memory, with the 8085 microprocessor.

Enhance problem-solving skills and the ability to analyse and debug programs written for the 8085 microprocessor.

Overall, the objective is to provide students with a practical understanding of the 8085 architecture and its applications in computer systems, enabling them to design and implement basic microprocessor-based systems.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Examine computer architecture**':

1. Competency in understanding microprocessor architecture: Students will gain a solid understanding of the architecture and organization of the 8085 microprocessor, including its various components, registers, and data transfer mechanisms.

D. Relevant Course Outcomes (COs):

1. Examine 8085 Architecture and its working.

E. Practical Outcomes:

1. Microcontroller Programming: Understanding the 8085 architecture prepares individuals to work with similar microcontrollers and microprocessors, enabling them to program and control these devices efficiently.
2. Embedded Systems Development: Knowledge of 8085 architecture provides a foundation for designing and developing embedded systems that utilize microcontrollers. This includes projects such as home automation systems, robotics, industrial control systems, and more.

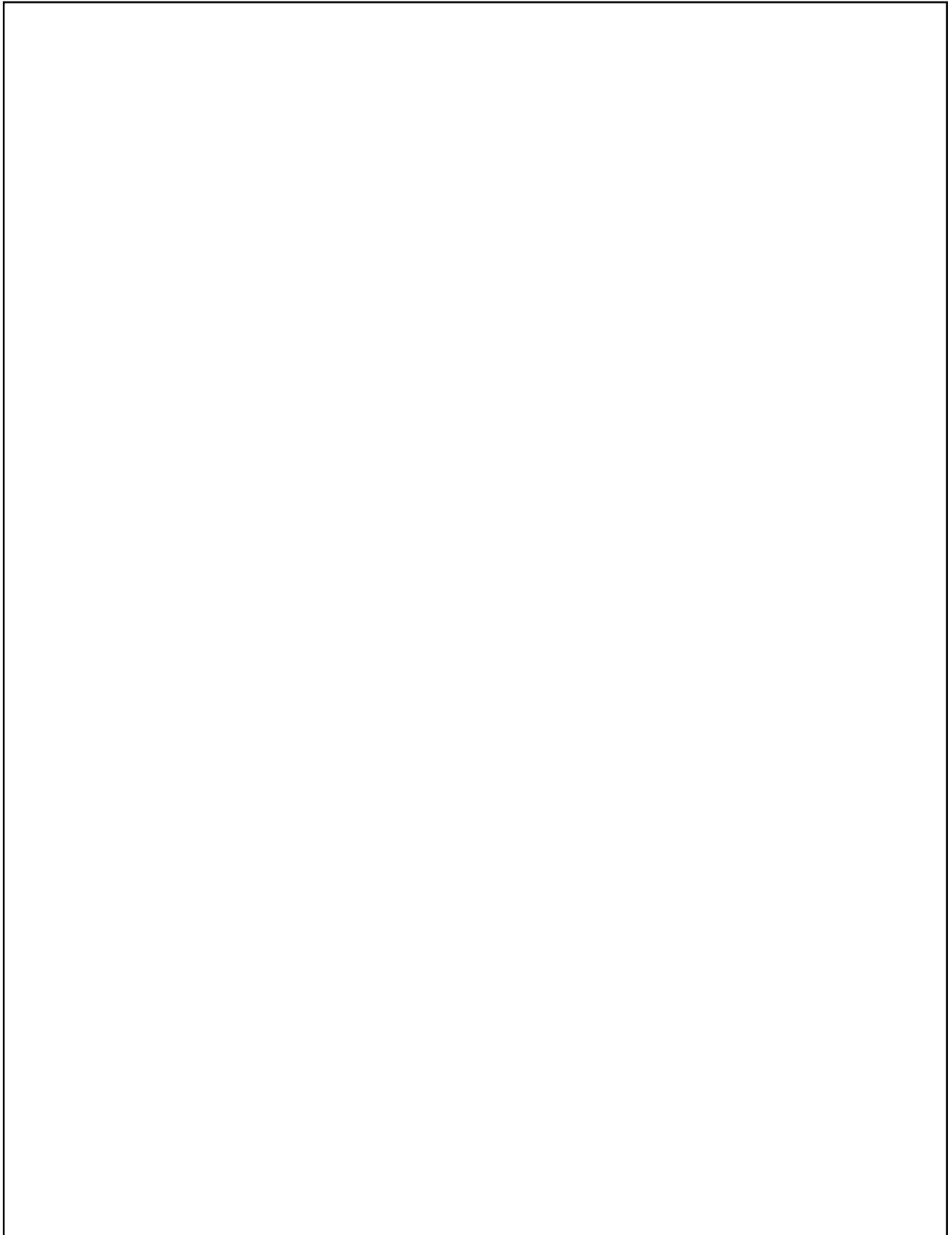
F. Relevant Affective domain Outcomes (ADOs):

1. Appreciation for Technical Detail
2. Curiosity and Inquisitiveness
3. Sense of Achievement
4. Passion for Technology.

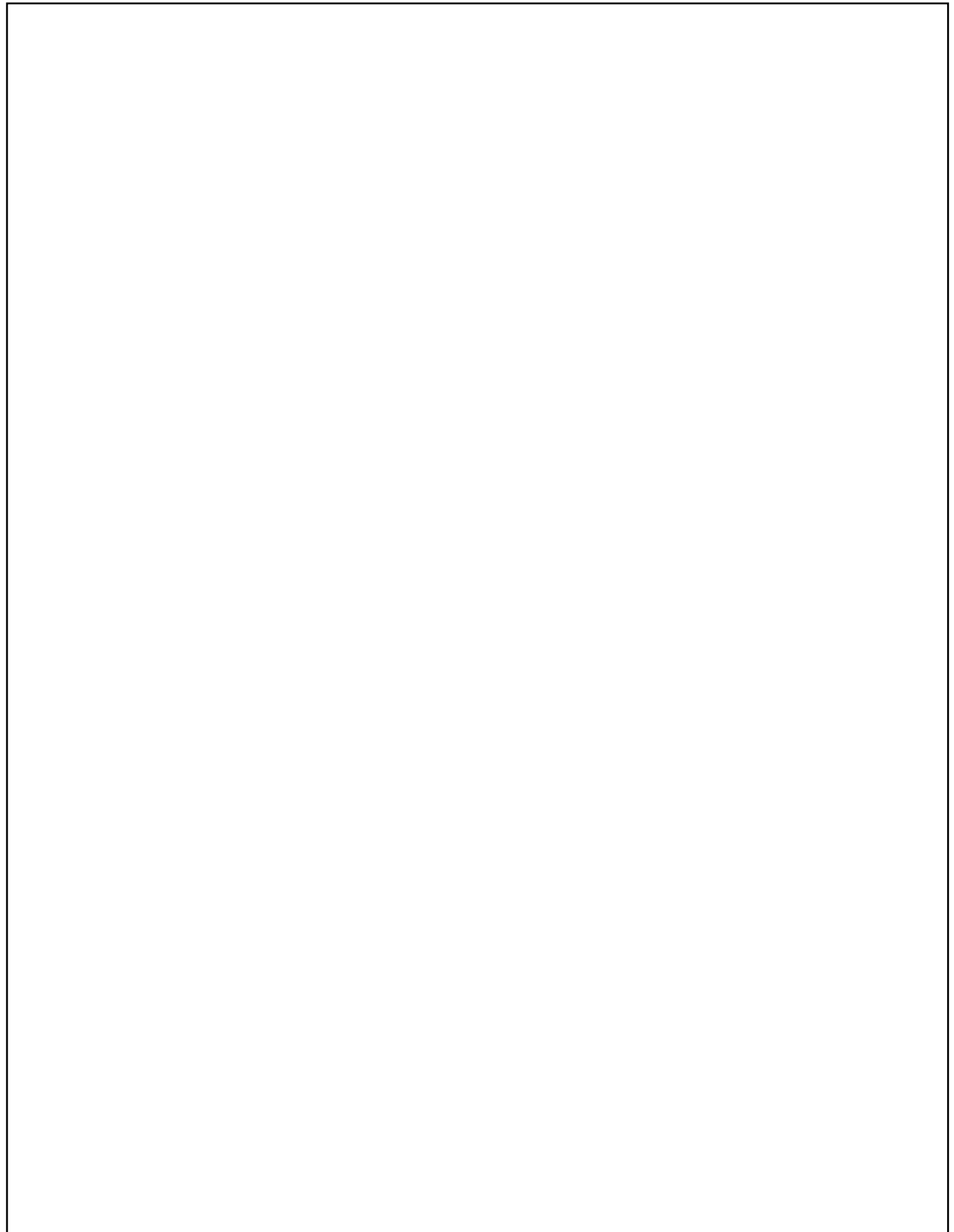
G. Prerequisite Theory:

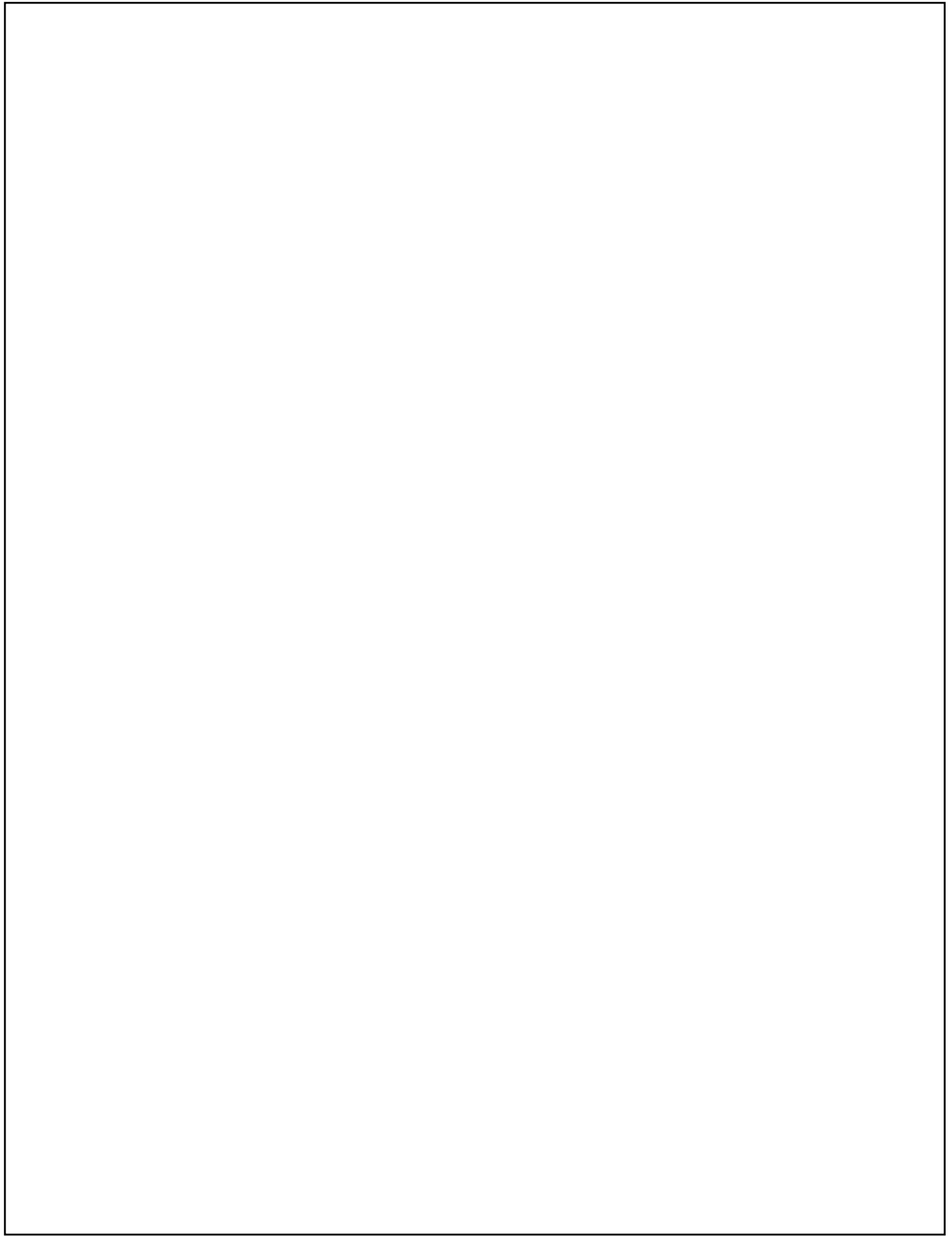
- Digital Logic: Understanding digital logic gates, Boolean algebra, and basic digital circuit design concepts helps in comprehending the internal workings of the microprocessor.
- Number Systems: Familiarity with number systems, especially binary and hexadecimal, is essential for understanding the representation of data and instructions in the 8085 microprocessor.
- Computer Organization: Knowledge of computer organization principles, including CPU, memory, input/output devices, and the fetch-decode-execute cycle, provides a foundation for understanding the overall structure of the 8085 microprocessor.
- Basic Electronics: Understanding basic electronic components and circuits, such as resistors, capacitors, transistors, and flip-flops, is helpful in understanding the interface and interaction of the microprocessor with the external world

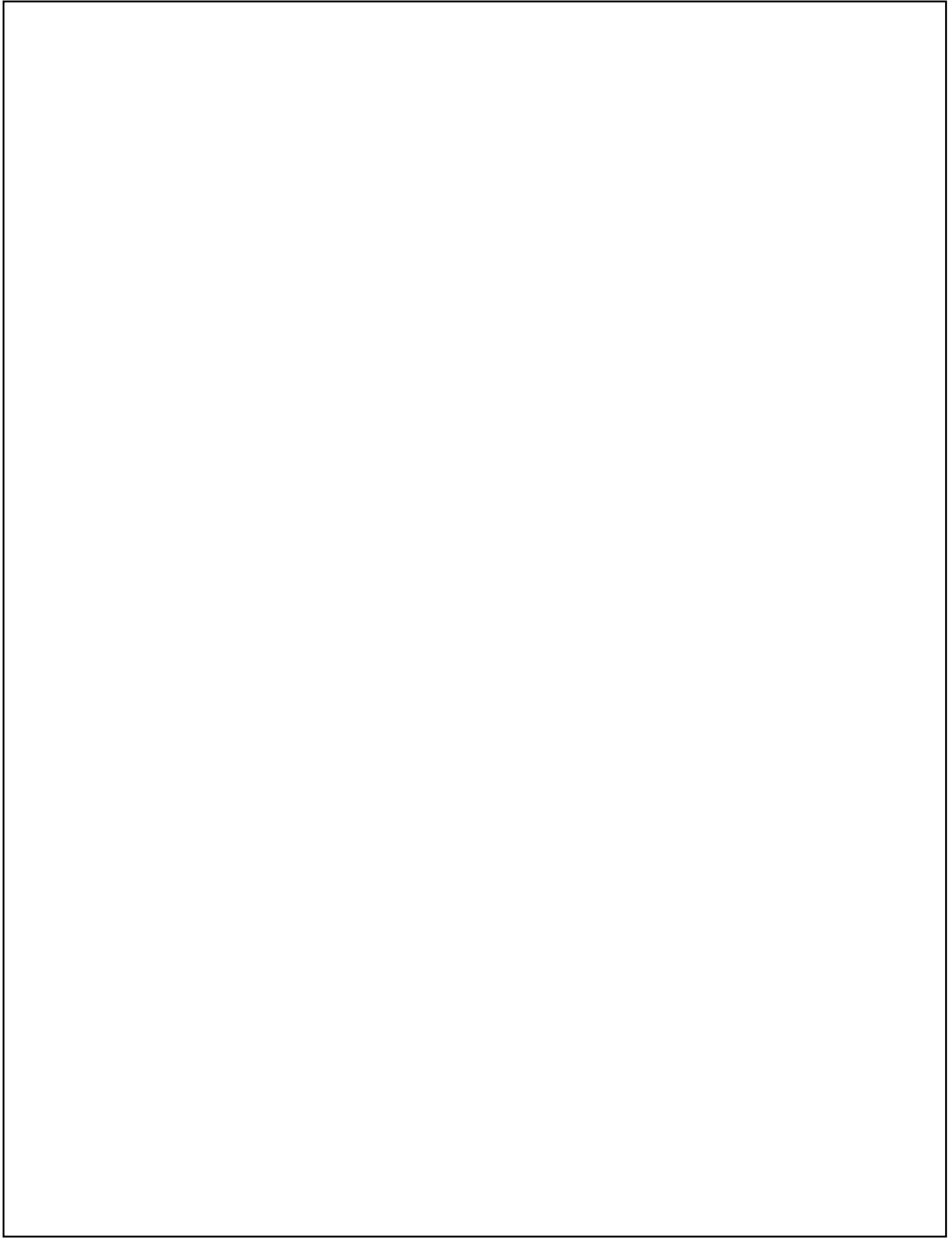
H. Diagram of 8085 Architecture

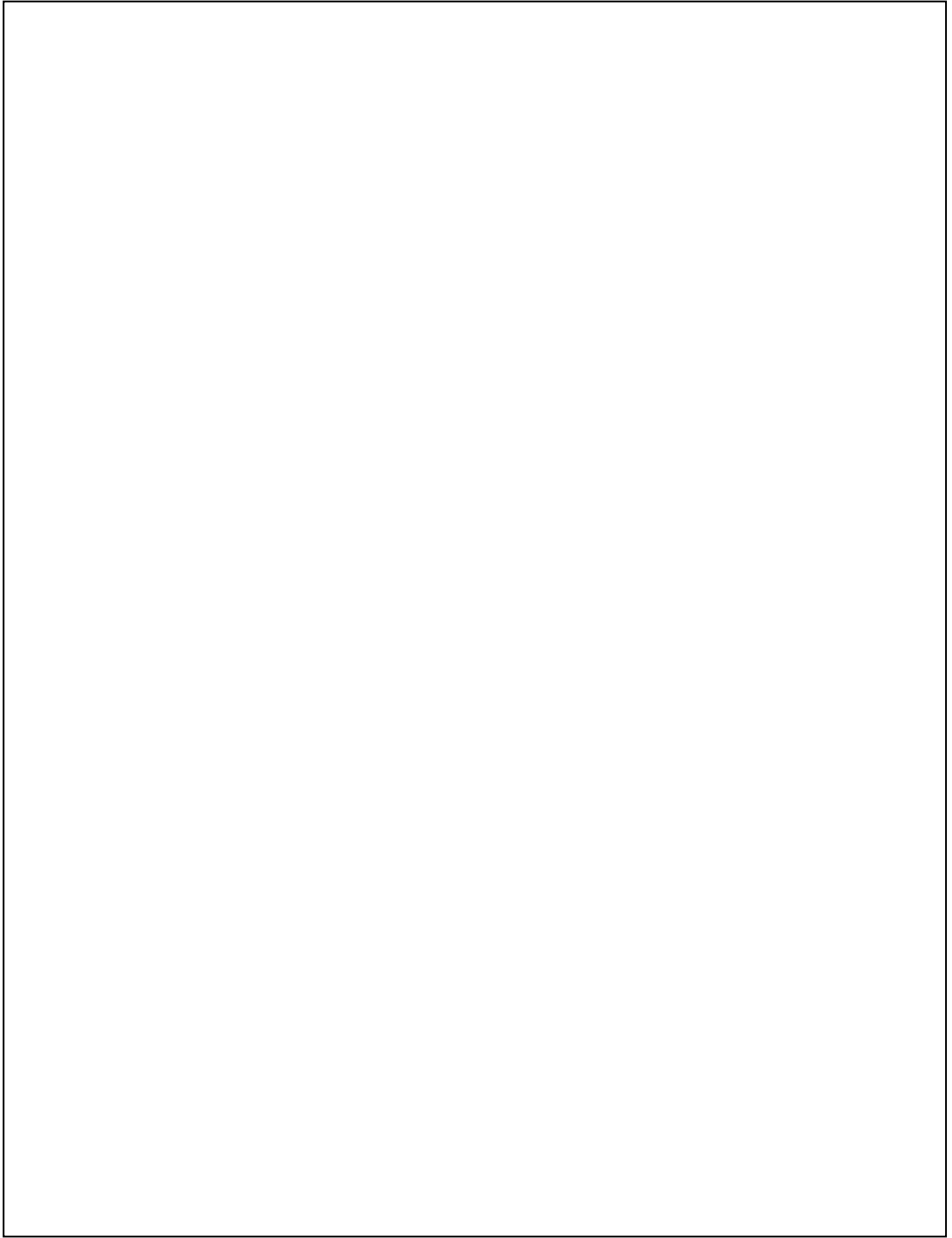


I. Description of Various Components of 8085 Architecture









J. Practical related Quiz:

1. What is the primary function of the ALU (Arithmetic Logic Unit) in the 8085 microprocessor?
 - a) Data storage
 - b) Arithmetic and logical operations
 - c) Instruction fetching
 - d) Memory addressing
2. Which component of the 8085 microprocessor is responsible for storing temporary data during processing?
 - a) Accumulator
 - b) Program Counter
 - c) Register File
 - d) Stack Pointer
3. Which bus connects the ALU, registers, and internal data paths in the 8085 microprocessor?
 - a) Address bus
 - b) Data bus
 - c) Control bus
 - d) Power bus
4. The instruction decoder in the 8085 microprocessor is responsible for:
 - a) Fetching instructions from memory
 - b) Executing arithmetic operations
 - c) Storing data temporarily
 - d) Controlling the flow of instructions
5. Which component of the 8085 microprocessor holds the memory address of the next instruction to be fetched?
 - a) Stack Pointer
 - b) Instruction Register
 - c) Program Counter
 - d) Memory Address Register
6. The control signals generated by the control unit in the 8085 microprocessor are used for:
 - a) Addressing memory locations
 - b) Performing arithmetic operations
 - c) Controlling data transfer between registers
 - d) Managing the overall operation of the microprocessor

7. The interrupt lines in the 8085 microprocessor are used for:
 - a) Communicating with external devices
 - b) Controlling the clock frequency
 - c) Storing intermediate results
 - d) Performing logical operations
8. Which component of the 8085 microprocessor is responsible for temporarily storing the return address of subroutines?
 - a) Accumulator
 - b) Stack Pointer
 - c) Program Counter
 - d) Instruction Register
9. The control bus in the 8085 microprocessor is used for:
 - a) Transmitting data between the microprocessor and memory
 - b) Controlling the flow of instructions
 - c) Carrying out arithmetic and logical operations
 - d) Addressing memory locations
10. The function of the instruction register in the 8085 microprocessor is to:
 - a) Store the result of arithmetic operations
 - b) Store the opcode of the current instruction
 - c) Store the address of the next instruction
 - d) Store the data to be transferred to memory

Answers:

- b) Arithmetic and logical operations
- c) Register File
- b) Data bus
- a) Fetching instructions from memory
- c) Program Counter
- d) Managing the overall operation of the microprocessor
- a) Communicating with external devices
- b) Stack Pointer
- b) Controlling the flow of instructions
- b) Store the opcode of the current instruction

Signature

Practical No. 03: Data Transfer Instructions

1. Summarize out Data Transfer Instructions and perform minimum 3 to 5 programs associated with the said concept.

A. Objectives:

Understand Data Transfer Instructions: Familiarize students with the various data transfer instructions available in the assembly language of the given microprocessor (such as the 8085).

Gain Hands-on Experience: Provide students with practical exposure to writing assembly language programs that involve data transfer instructions. This helps them develop proficiency in using these instructions effectively.

Learn Data Movement Techniques: Explore different techniques and methods for moving data within the microprocessor's registers, memory locations, and I/O devices using the data transfer instructions.

Develop Programming Skills: Enhance students' programming skills by practicing the implementation of data transfer instructions in assembly language. This includes understanding addressing modes, syntax, and the overall structure of assembly programs.

Analyse and Debug Programs: Enable students to analyse, debug, and troubleshoot assembly language programs that involve data transfer instructions. This helps them develop critical thinking and problem-solving skills in the context of microprocessor programming.

Apply Knowledge to Real-world Scenarios: Encourage students to apply their understanding of data transfer instructions to practical scenarios and real-world applications. This could include tasks such as data copying, data manipulation, and interfacing with external devices.

Reinforce Understanding of Computer Organization: Strengthen students' understanding of computer organization concepts, such as memory hierarchies, register utilization, and data movement between different components of the microprocessor.

Document and Report: Develop skills in documenting and reporting the assembly language programs, including program descriptions, flowcharts, and expected outcomes. This promotes effective communication and presentation of the implemented solutions.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Engineering Tools, Experimentation and Testing (PO4):** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.

- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Assembly Language programs involving Data Transfer Instructions**':

1. Programming skills.
2. Debugging skills.
3. Hardware Understanding

D. Relevant Course Outcomes (COs):

1. Perform Assembly language programming using 8085 Instruction Set.

E. Practical Outcomes:

1. Develop assembly language programs involving Data Transfer Instructions in 8085 microprocessors.

F. Relevant Affective domain Outcomes (ADOs):

1. Appreciation for Low-level Programming.
2. Curiosity and Interest in Assembly Language.
3. Confidence in Programming.
4. Critical Thinking and Troubleshooting Skills.
5. Follow ethical practices

G. Prerequisite Theory:

Instruction	Usage	Register Transfer Language	Brief Description	Instruction Size	Addressing Mode
Move Instructions					
MOV	MOV Rd, Rs	Rd <- Rs	Move data from the source register (Rs) to the destination register (Rd).	1 byte	Register
MVI	MVI Rd, data	Rd <- data	Move immediate data (8-bit) into the destination register (Rd).	2 bytes	Immediate
Load & Store Instructions					
LDA	LDA addr	A <- (addr)	Load the accumulator (A) with data from the memory location specified by the address (addr).	3 bytes	Direct
STA	STA addr	(addr) <- A	Store the content of the accumulator (A) into the memory location specified by the address (addr).	3 bytes	Direct
LDAX	LDAX Rp	A <- (Rp)	Load the accumulator (A) with data from the memory location pointed by the register pair (Rp).	1 byte	Register Pair
STAX	STAX Rp	(Rp) <- A	Store the content of the accumulator (A) into the memory location pointed by the register pair (Rp).	1 byte	Register Pair
LHLD	LHLD addr	L <- (addr); H <- (addr+1)	Load the L and H registers with data from consecutive memory locations specified by the address (addr).	3 bytes	Direct
SHLD	SHLD addr	(addr) <- L; (addr+1) <- H	Store the content of the L and H registers into consecutive memory locations specified by the address (addr).	3 bytes	Direct
Load Register Pair					
LXI	LXI Rp, data	Rp <- data (16-bit)	Load the register pair (Rp) with immediate 16-bit data.	3 bytes	Immediate

Exchange Instructions					
XCHG	XCHG	H \leftrightarrow D; L \leftrightarrow E	Exchange the content of the H and L registers with the content of the D and E registers.	1 byte	Implicit
XTHL	XTHL	L \leftrightarrow (SP); H \leftrightarrow (SP+1)	Exchange the content of the H and L registers with the data from the top of the stack.	1 byte	Implicit
SPHL	SPHL	SP \leftarrow HL	Copy the content of the H and L registers into the stack pointer (SP).	1 byte	Implicit
Stack Instructions					
PUSH	PUSH Rp	(SP-1) \leftarrow RpH; (SP-2) \leftarrow RpL; SP \leftarrow SP-2	Push the content of the register pair (Rp) onto the stack.	1 byte	Register Pair
PUSH	PUSH PSW	(SP-1) \leftarrow A; (SP-2) \leftarrow Flags; SP \leftarrow SP-2	Push the content of the accumulator (A) and the flags register onto the stack.	1 byte	Implicit
POP	POP Rp	RpL \leftarrow (SP); RpH \leftarrow (SP+1); SP \leftarrow SP+2	Pop the content from the top of the stack into the register pair (Rp).	1 byte	Register Pair
POP	POP PSW	Flags \leftarrow (SP); A \leftarrow (SP+1); SP \leftarrow SP+2	Pop the content from the top of the stack into the accumulator (A) and the flags register.	1 byte	Implicit
PCHL Instruction					
PCHL	PCHL	PC \leftarrow HL	Load the program counter (PC) with the content of the H and L registers.	1 byte	Implicit

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: Dual Core RAM: 4 GB Operating System: Windows 7
2.	GNU 8085 Simulator Software	

I. Source code and Output:

Write an assembly language program to copy data from memory location 2050h to 3050h.

Before Execution:

[2050h]: 32h

Source Code:

After Execution:

[3050h]:

Write an assembly language program to swap data on memory location 2050h & 3050h.

Before Execution:

[2050h]: 32h

[3050h]: 56h

Source Code:

After Execution:

[2050h]: _____

[3050h]: _____

Write an assembly language program to Load data on 2050h into Accumulator and Store it back on Location 3050h.

Before Execution:

[2050h]: 32h

Source Code:

After Execution:

[2050h]: 32h

[3050h]: 56h

[A] : _____

Write an assembly language program to move immediate data 96h into Register B and then move Immediate data 43h to register C. Finally load the HL pair with the 16-bit data stored at memory location 9643h.

Before Execution:

[9643h]: 32h

[9644h]: 85h

Source Code:

After Execution:

[BC]: _____

[HL]: _____

Write an assembly language program to load data into accumulator which is pointed by memory location 1000h, also store that data into 2000h memory location.

Before Execution:

[1000h]: 59h

[1001h]: 32h

[3259h]: 72h

Source Code:

After Execution:

[A] : _____

[2000h]: _____

Write an assembly language program to load accumulator with input data from port 78h and also give that data to the output device connected on port 39h

Before Execution:

[78h]: 23h

Source Code:

After Execution:

[A] : _____

[39h]: _____

J. Practical related Quiz:

1. What is the purpose of the 8085 MOV instruction?
 - a) Move data from memory to accumulator
 - b) Move data from accumulator to memory
 - c) Move data between registers
 - d) Perform arithmetic addition
2. Which addressing mode is used by the LXI instruction in the 8085 microprocessor?
 - a) Direct addressing mode
 - b) Immediate addressing mode
 - c) Register addressing mode
 - d) Indirect addressing mode
3. What does the instruction "MVI B, 0AH" do in the 8085 microprocessor?
 - a) Move immediate data 0AH to register B
 - b) Move register B to register A
 - c) Move data from memory address 0AH to register B
 - d) Move data from register B to memory address 0AH
4. The XCHG instruction in the 8085 microprocessor is used to:
 - a) Exchange the content of register pair HL with the content of register pair DE
 - b) Exchange the content of the accumulator with the content of memory location 0
 - c) Exchange the content of register pair BC with the content of register pair HL
 - d) Exchange the content of the stack pointer with the content of the program counter
5. What is the size (in bytes) of the LDA instruction in the 8085 microprocessor?
 - a) 1 byte
 - b) 2 bytes
 - c) 3 bytes
 - d) 4 bytes

6. The LHLD instruction in the 8085 microprocessor is used to load data from:
 - a) Register pair HL to memory location specified by the address
 - b) Memory location specified by the address to register L
 - c) Memory location specified by the address to register H
 - d) Accumulator to memory location specified by the address
7. Which data transfer instruction is used to push the content of register pair BC onto the stack in the 8085 microprocessors?
 - a) PUSH B
 - b) PUSH BC
 - c) PUSH C
 - d) PUSH H
8. What does the 8085 instruction "STAX H" do?
 - a) Store the content of register H at the memory location specified by the address stored in the stack pointer (SP).
 - b) Store the content of register H at the memory location specified by the address stored in the program counter (PC).
 - c) Store the content of register H at the memory location specified by the content of the accumulator (A).
 - d) Store the content of register H at the memory location specified by the content of the HL register pair.
9. What is the purpose of the XTHL instruction in the 8085 microprocessor?
 - a) Exchange the content of the stack pointer with the content of register pair HL
 - b) Exchange the content of the accumulator with the content of register H
 - c) Exchange the content of register pair HL with the content of memory location 0
 - d) Exchange the content of register pair BC with the content of register pair DE
10. The instruction "PCHL" in the 8085 microprocessor is used to:
 - a) Load the program counter with the content of register pair HL
 - b) Load the program counter with the content of register pair BC
 - c) Load the program counter with the content of register pair DE
 - d) Load the program counter with the content of the accumulator

Answers

- c) Move data between registers
- b) Immediate addressing mode
- a) Move immediate data 0AH to register B
- a) Exchange the content of register pair HL with the content of register pair DE
- c) 3 bytes
- a) Register pair HL to memory location specified by the address
- b) PUSH BC
- d) Store the content of register H at the memory location specified by the content of the HL register pair.
- a) Exchange the content of the stack pointer with the content of register pair HL
- a) Load the program counter with the content of register pair HL

Signature

Practical No. 04: Arithmetic Instructions

1. Summarize Arithmetic Instructions and perform minimum 3 to 5 programs associated with the said concept

A. Objectives:

Understand Arithmetic Instructions: Gain a clear understanding of various arithmetic instructions, such as ADD, SUB, INR, DCR, ADC, SBB, and others, and their usage in the 8085 microprocessors.

Develop Assembly Language Programming Skills: Enhance programming skills in assembly language and learn to write efficient code using arithmetic instructions for the 8085 microprocessors.

Implement Practical Arithmetic Operations: Practice Register and Memory Manipulation: Practice manipulating data stored in registers and memory locations to perform arithmetic operations using arithmetic instructions.

Learn Flag Manipulation: Understand how arithmetic instructions affect the flag register and how flags are used to interpret the results of arithmetic operations.

Apply arithmetic instructions to perform practical arithmetic operations, such as addition, subtraction, multiplication, and division, on data stored in the 8085 microprocessors.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Engineering Tools, Experimentation and Testing (PO4):** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.
- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Develop Assembly Language programs involving Arithmetic Instructions**’:

1. Programming skills.
2. Debugging skills.
3. Hardware Understanding

D. Relevant Course Outcomes (COs):

1. Perform Assembly language programming using 8085 Instruction Set.

E. Practical Outcomes:

1. Develop assembly language programs involving Arithmetic Instructions in 8085 microprocessors.

F. Relevant Affective domain Outcomes (ADOs):

1. Appreciation for Low-level Programming.
2. Curiosity and Interest in Assembly Language.
3. Confidence in Programming.
4. Critical Thinking and Troubleshooting Skills.
5. Follow ethical practices

G. Prerequisite Theory:

Instruction	Usage	Register Transfer Language	Brief Description	Instruction Size	Addressing Mode
Add Instructions					
ADD	ADD R/M	$A \leftarrow A + R/M$	Adds the content of register R to the accumulator A	1 byte	Register
ADI	ADI 8-bit data	$A \leftarrow A + 8\text{-bit data}$	Adds the 8-bit immediate data to the accumulator A	2 bytes	Immediate
ADC	ADC R/M	$A \leftarrow A + R/M + CY$	Adds the content of register R and carry flag to the accumulator A	1 byte	Register
ACI	ACI 8-bit data	$A \leftarrow A + 8\text{-bit data} + CY$	Adds the 8-bit immediate data and carry flag to the accumulator A	2 bytes	Immediate
DAD	DAD Rp	$HL \leftarrow HL + Rp$	Adds the content of register pair Rp to HL (Rp can be BC, DE, or HL)	1 byte	Register

Subtract Instructions					
SUB	SUB R/M	$A \leftarrow A - R/M$	Subtracts the content of register R from the accumulator A	1 byte	Register
SUI	SUI 8-bit data	$A \leftarrow A - \text{8-bit data}$	Subtracts the 8-bit immediate data from the accumulator A	2 bytes	Immediate
SBB	SBB R/M	$A \leftarrow A - R/M - CY$	Subtracts the content of register R and carry flag from the accumulator A	1 byte	Register
SBI	SBI 8-bit data	$A \leftarrow A - \text{8-bit data} - CY$	Subtracts the 8-bit immediate data and carry flag from the accumulator A	2 bytes	Immediate
Increment/Decrement Instructions					
INR	INR R/M	$R \leftarrow R/M + 1$	Increments the content of register R by 1	1 byte	Register
DCR	DCR R/M	$R \leftarrow R/M - 1$	Decrements the content of register R by 1	1 byte	Register
INX	INX Rp	$R_p \leftarrow R_p + 1$	Increments the content of register pair Rp by 1 (Rp can be BC, DE, or HL)	1 byte	Register
DCX	DCX Rp	$R_p \leftarrow R_p - 1$	Decrements the content of register pair Rp by 1 (Rp can be BC, DE, or HL)	1 byte	Register
Decimal Adjust Accumulator (DAA)					
DAA	DAA	$A \leftarrow \text{BCD Corrected A}$	Adjusts the result in the accumulator after BCD addition or subtraction	1 byte	No Operand

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: Dual Core RAM: 4 GB Operating System: Windows 7
2.	GNU 8085 Simulator Software	

I. Source code and Output:

Write an assembly language program to add two numbers stored at memory locations 2000h and 2001h and also store the result back into memory location 2002h.

Before Execution:

[2000h]: _____

[2001h]: _____

Source Code:**After Execution:**

[2002h]: _____

Write an assembly language program to subtract two numbers stored at memory locations 1000H and 1001H and also store the result back into memory location 1002H..

Before Execution:

[1000h]: _____

[1001h]: _____

Source Code:

After Execution:

[1002h]: ____

Write an assembly language program to increment a number stored in memory location 8000h and store the result back into the same memory location using indirect addressing mode.

Before Execution:

[8000h]: 32h

Source Code:

After Execution:

[8001h]: ____

Write an assembly language program to decrement a 16-bit number stored in memory location 8000h and store the result back into the same memory location.

Before Execution:

[8000h]: 1000h

Source Code:

After Execution:

[8000h]: _____

Write an assembly language program to add 2 16-bit numbers stored in memory locations 2050h and 2052h and store the answer at memory location 2054h.

Before Execution:

[2050h]: _____

[2052h]: _____

Source Code:

After Execution:

[2054h]: _____

Write an assembly language program to add 2 16-bit numbers stored in memory locations 2050h and 2052h without using DAD instruction and store the answer at memory location 2054h.

Before Execution:

[2050h]: _____

[2052h]: _____

Source Code:

After Execution:

[2054h]: _____

J. Practical related Quiz:

1. Which 8085 arithmetic instruction is used to add the content of register R to the accumulator?
 - a) ADD R
 - b) SUB R
 - c) ADC R
 - d) INR R
2. Which instruction is used to subtract the 8-bit immediate data from the accumulator in the 8085 microprocessors?
 - a) ADD
 - b) SUB
 - c) ACI
 - d) SUI
3. The instruction DAA (Decimal Adjust Accumulator) is used to:
 - a) Perform arithmetic operations on 16-bit data
 - b) Adjust the result in the accumulator after BCD addition or subtraction
 - c) Multiply two numbers
 - d) Perform logical operations
4. What does the instruction INR R do in the 8085 microprocessors?
 - a) Increments the content of register R by 1
 - b) Decrements the content of register R by 1
 - c) Adds the content of register R to the accumulator
 - d) Subtracts the content of register R from the accumulator
5. The instruction MOV A, B in the 8085 microprocessor means:
 - a) Move the content of register B to the accumulator
 - b) Move the content of accumulator A to register B
 - c) Add the content of register B to the accumulator
 - d) Subtract the content of register B from the accumulator

6. Which of the following instructions is used to add the 8-bit immediate data and carry flag to the accumulator in the 8085 microprocessors?
 - a) ACI
 - b) ADD
 - c) ADC
 - d) SBB
7. The instruction DCR R in the 8085 microprocessors:
 - a) Decrements the content of register R by 1
 - b) Increments the content of register R by 1
 - c) Adds the content of register R to the accumulator
 - d) Subtracts the content of register R from the accumulator
8. The instruction DAD R in the 8085 microprocessor is used to:
 - a) Add the content of register R to the accumulator
 - b) Add the content of register pair R to the HL register pair
 - c) Subtract the content of register R from the accumulator
 - d) Subtract the content of register pair R from the HL register pair
9. Which instruction is used to subtract the content of register R and carry flag from the accumulator in the 8085 microprocessors?
 - a) ACI
 - b) ADD
 - c) ADC
 - d) SBB
10. The instruction SBI 8-bit data in the 8085 microprocessor means:
 - a) Subtract the 8-bit immediate data from the accumulator and borrow
 - b) Subtract the 8-bit immediate data from the accumulator and carry
 - c) Subtract the 8-bit immediate data from the accumulator
 - d) Subtract the 8-bit immediate data from the accumulator and add carry

Answers:

- a) ADD R
- d) SUI
- b) Adjust the result in the accumulator after BCD addition or subtraction
- a) Increments the content of register R by 1
- a) Move the content of register B to the accumulator
- a) ACI
- a) Decrements the content of register R by 1
- b) Add the content of register pair R to the HL register pair
- d) SBB
- a) Subtract the 8-bit immediate data from the accumulator and borrow

Signature

Practical No. 05: Logical Instructions

1. Summarize Logical Instructions of 8085 with example and execute minimum 3 to 5 programs associated with said concept.

A. Objectives:

Understand Logical Instructions: Students will learn about various logical instructions available in the microprocessor's instruction set, including AND, OR, XOR, and NOT instructions. They will grasp the significance of these instructions in manipulating data at the bit level.

Perform Bitwise Operations: Students will practice using logical instructions to perform bitwise operations on data stored in registers and memory locations. They will learn how to apply these operations to specific bits within binary numbers.

Master Binary Calculations: Through hands-on programming exercises, students will gain proficiency in using logical instructions to perform binary calculations, such as binary addition and subtraction.

Check Conditions: Logical instructions are often used to check specific conditions in data. Students will learn how to utilize AND, OR, and NOT instructions to test and modify specific bits to evaluate conditions.

Set and Clear Flags: The practical will involve using logical instructions to set or clear flags based on specific conditions. Students will understand the importance of flags in the control flow of a program.

Implement Shift Instructions: Students will practice using shift instructions (e.g., SHL, SHR) to perform left and right shifts on data, which are essential for multiplication and division operations in binary arithmetic.

Enhance Programming Proficiency: The practical will improve students' programming proficiency by introducing them to a broader range of instructions and expanding their understanding of assembly language programming.

Foster Appreciation for Low-Level Programming: Students will develop an appreciation for low-level programming and its significance in optimizing code execution and improving system performance.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Engineering Tools, Experimentation and Testing (PO4):** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.

- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Assembly Language programs involving Logical Instructions**':

1. Programming skills.
2. Debugging skills.
3. Hardware Understanding

D. Relevant Course Outcomes (COs):

1. Perform Assembly language programming using 8085 Instruction Set.

E. Practical Outcomes:

1. Develop assembly language programs involving Arithmetic Instructions in 8085 microprocessors.

F. Relevant Affective domain Outcomes (ADOs):

1. Appreciation for Low-level Programming.
2. Curiosity and Interest in Assembly Language.
3. Confidence in Programming.
4. Critical Thinking and Troubleshooting Skills.
5. Follow ethical practices

G. Prerequisite Theory:

Instruction	Usage	Register Transfer Language	Brief Description	Instruction Size	Addressing Mode
AND Instructions					
ANA	ANA R/M	A <- A & R/M	Logical AND accumulator with R/M	1 byte	Register
ANI	ANI 8-bit	A <- A & 8-bit data	Logical AND accumulator with 8-bit data	2 bytes	Immediate

OR Instructions					
ORA	ORA R/M	$A \leftarrow A \mid R/M$	Logical OR accumulator with R/M	1 byte	Register
ORI 8-bit	0xF6	$A \leftarrow A \mid 8\text{-bit data}$	Logical OR accumulator with 8-bit data	2 bytes	Immediate
EX-OR Instructions					
XRA	XRA R/M	$A \leftarrow A \wedge R/M$	Logical EXCLUSIVE OR accumulator with R/M	1 byte	Register
XRI	XRI 8-bit	$A \leftarrow A \wedge 8\text{-bit data}$	Logical EXCLUSIVE OR accumulator with 8-bit data	2 bytes	Immediate
Set/Complement Instructions					
CMA	CMA	$A \leftarrow !A$	Complement (NOT) accumulator	1 byte	None
CMC	CMC	$CY \leftarrow !CY$	Complement (NOT) carry flag	1 byte	None
STC	STC	$CY \leftarrow 1$	Set carry flag	1 byte	None
Compare Instructions					
CMP	CMP R/M	$A - R/M$	Compare accumulator with R/M If $CY = 1, Z = 0$ then $A > R/M$ If $CY = 0, Z = 1$ then $A = R/M$ If $CY = 0, Z = 0$ then $A < R/M$	1 byte	Register
CPI	CPI 8-bit	$A - 8\text{-bit data}$	Compare accumulator with 8-bit data	2 bytes	Immediate
Rotate Instructions					
RLC	RLC	$A = A \ll 1, CY = \text{MSB of } A$	Rotate accumulator left through carry	1 byte	None
RRC	RRC	$A = A \gg 1, CY = \text{LSB of } A$	Rotate accumulator right through carry	1 byte	None

RAL	RAL	$A = A \ll 1$, CY = MSB of A, MSB = CY	Rotate accumulator left through carry with MSB = CY	1 byte	None
RAR	RAR	$A = A \gg 1$, CY = LSB of A, MSB = CY	Rotate accumulator right through carry with MSB = CY	1 byte	None

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: Dual Core RAM: 4 GB Operating System: Windows 7
2.	GNU 8085 Simulator Software	

I. Source code and Output:

Write an assembly language program to mask the lower nibble (4 bits) of the data stored at memory location 2050h. Also store the result on 2051h.

Before Execution:

[2050h]: ____

Source Code:

After Execution:

[2051h]: ____

Write an assembly language program to set odd bits of the data stored at 1000h and store the answer back on 1001h.

Before Execution:

[1000h]: ____

Source Code:

After Execution:

[1001h]: ____

Write an assembly language program to multiply the number stored in the location pointed by HL register pair by 4. Store the answer in 1000h.

Before Execution:

[H][L]: 3592h

[3592h]: 15h

Source Code:

After Execution:

[1000h]: ____

Write an assembly language program to clear the contents of an accumulator and set it to 00h without using any data transfer instruction.

Before Execution:

[A]: 12h

Source Code:

After Execution:

[A]: ____

Write an assembly language program to check the bit – 3 (4th bit) of the number stored in memory location 1000h. The answer can be observed in Zero Flag (Z).

Before Execution:

[1000h]: 59h

Source Code:

After Execution:

[ZF]: ____

Write an assembly language program to generate 2's Complement of the number stored in Register B and store the answer back in Register C.

Before Execution:

[B]: 47h

Source Code:

After Execution:

[C]: _____

J. Practical related Quiz:

1. What does the 8085 instruction "ANA A" perform?
 - a. Logical AND operation between accumulator and itself
 - b. Logical OR operation between accumulator and itself
 - c. Logical XOR operation between accumulator and itself
 - d. Logical AND operation between accumulator and register A
2. Which instruction is used to mask the lower nibble of the data stored at memory location 2050h?
 - a. ANA 0F0h
 - b. ORA 0FFh
 - c. ANI 0Fh
 - d. XRI 00Fh
3. The 8085 instruction "XRA C" performs what operation?
 - a. Logical XOR between accumulator and register C
 - b. Logical AND between accumulator and register C
 - c. Logical OR between accumulator and register C
 - d. Logical NOT operation on register C
4. What is the size (in bytes) of the instruction "ANI 23h"?
 - a. 1 byte
 - b. 2 bytes
 - c. 3 bytes
 - d. 4 bytes
5. Which logical instruction is used to set specific bits in a register based on a given bitmask?
 - a. ANA
 - b. ORA
 - c. XRA
 - d. CMA
6. What is the result of "ANA 00110110b" if the accumulator contains "11001100b"?
 - a. 10000000b
 - b. 11111111b
 - c. 00000000b
 - d. 00110110b

7. The 8085 instruction "XRA 28h" performs what operation?
 - a. Logical OR between accumulator and 28h
 - b. Logical AND between accumulator and 28h
 - c. Logical XOR between accumulator and 28h
 - d. Logical NOT operation on 28h
8. Which logical instruction can be used to toggle specific bits in a register?
 - a. ANA
 - b. ORA
 - c. XRA
 - d. CMA
9. What is the result of "ANI 10011101b" if the accumulator contains "10101010b"?
 - a. 10101010b
 - b. 10011001b
 - c. 00000000b
 - d. 10011101b
10. The instruction "ORA 10110011b" performs what operation on the accumulator?
 - a. Logical AND with 10110011b
 - b. Logical OR with 10110011b
 - c. Logical XOR with 10110011b
 - d. Logical NOT operation

Answers:

- a) Logical AND operation between accumulator and itself
- a) ANA 0F0h
- a) Logical XOR between accumulator and register C
- b) 2 bytes
- b) ORA
- c) 00000000b
- c) Logical XOR between accumulator and 28h
- c) XRA
- d) 10011101b
- b) Logical OR with 10110011b

Signature

Practical No. 06: I/O Instructions

1. List Input-Output Instructions of 8085 with example and execute minimum 3 to 5 programs associated with said concept.

A. Objectives:

Introduction to I/O Operations: Introduce students to input and output instructions in 8085 microprocessors, enabling them to interact with external devices.

Understanding I/O Ports: Familiarize students with I/O ports and their addresses, illustrating how data is read from and written to these ports.

Interfacing with External Devices: Demonstrate the process of connecting the microprocessor to external devices such as sensors, displays, and LEDs for real-world applications.

Input Operations: Teach students how to use input instructions (IN) to read data from input ports, enabling them to receive external signals or data.

Output Operations: Educate students on using output instructions (OUT) to write data to output ports, allowing them to control external devices and displays.

Data Transfer: Provide hands-on experience in transferring data between input and output ports, highlighting the importance of data manipulation during I/O.

I/O Interfacing with Peripherals: Illustrate how I/O instructions play a crucial role in interfacing with peripheral devices, enabling students to understand the broader implications of I/O operations in embedded systems.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Engineering Tools, Experimentation and Testing (PO4):** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.
- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Develop Assembly Language programs involving Input-Output Instructions**’:

1. Programming skills.
2. Debugging skills.
3. Hardware Understanding

D. Relevant Course Outcomes (COs):

1. Perform Assembly language programming using 8085 Instruction Set.

E. Practical Outcomes:

1. Develop assembly language programs involving Input-Output Instructions in 8085 microprocessors.

F. Relevant Affective domain Outcomes (ADOs):

1. Appreciation for Low-level Programming.
2. Curiosity and Interest in Assembly Language.
3. Confidence in Programming.
4. Critical Thinking and Troubleshooting Skills.
5. Follow ethical practices

G. Prerequisite Theory:

Instruction	Usage	Register Transfer Language	Brief Description	Instruction Size	Addressing Mode
Input-Output Instructions					
IN	IN port	$A \leftarrow (\text{port})$	Input data from the specified input port (port) into the accumulator (A).	2 bytes	Immediate
OUT	OUT port	$(\text{port}) \leftarrow A$	Output the content of the accumulator (A) to the specified output port (port).	2 bytes	Immediate

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: Dual Core RAM: 4 GB Operating System: Windows 7
2.	GNU 8085 Simulator Software	

I. Source code and Output:

Write an assembly language program to read data from a specified input port (e.g., Port 26h) and store the data in the accumulator (A) for further processing.

Before Execution:

[26h]: _____

Source Code:**After Execution:**

[A]: _____

Write an assembly language program to output data from the accumulator (A) to a specified output port (e.g., Port 95h) for controlling external devices.

Before Execution:

[A]: _____

Source Code:**After Execution:**

[95h]: _____

Write an assembly language program to read data from one input port (e.g., Port 06h), perform some operations, and then write the result to an output port (e.g., Port 07h)..

Before Execution:

[06h]: ____

Source Code:

After Execution:

[07h]: ____

J. Practical related Quiz:

1. Which 8085 instruction is used to read data from an input port?
 - a) IN
 - b) OUT
 - c) ANA
 - d) XRA
2. The "OUT 05h" instruction in 8085 microprocessor will:
 - a) Read data from output port 05h
 - b) Write data to output port 05h
 - c) Read data from input port 05h
 - d) Write data to input port 05h
3. The "IN 08h" instruction in 8085 microprocessor will read data from:
 - a) Input port 08h
 - b) Output port 08h
 - c) Memory location 08h
 - d) Accumulator (A)
4. The "OUT 01h" instruction in 8085 microprocessor will:
 - a) Read data from output port 01h
 - b) Write data to output port 01h
 - c) Read data from input port 01h
 - d) Write data to input port 01h

5. Which instruction is used to write data to an output port?

- a) OUT
- b) IN
- c) MOV
- d) STA

Answers:

- a) IN
- b) Write data to output port 05h
- a) Input port 08h
- b) Write data to output port 01h
- a) OUT

Signature

Practical No. 07: Machine Control & Interrupt Instructions

1. Recall Machine Control & Interrupt Instructions of 8085 with example and execute minimum 2 to 3 programs associated with said concept.

A. Objectives:

Understanding Machine Control Instructions: Introduce students to the NOP (No Operation) and HLT (Halt) instructions in the 8085 microprocessors, providing insights into their roles in controlling the processor's behaviour.

Introduction to Interrupt Instructions: Familiarize students with the concept of interrupts in 8085 microprocessors, enabling them to comprehend the need for handling external events efficiently.

RIM and SIM Instructions Application: Educate students on using the RIM (Read Interrupt Mask) and SIM (Set Interrupt Mask) instructions to read and set the interrupt mask, respectively, understanding their significance in enabling or disabling interrupts.

Handling External Interrupts: Implement practical scenarios where the microprocessor responds to external interrupts, emphasizing the need for interrupt service routines (ISRs) to manage various interrupt sources.

Interrupt Priority Management: Illustrate the concept of interrupt priority management using RST instructions, enabling students to comprehend the priority levels of different interrupts.

Interrupt-Driven Data Transfer: Explore interrupt-driven data transfer techniques, showcasing the efficiency of interrupts in handling real-time data acquisition and processing tasks.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Engineering Tools, Experimentation and Testing (PO4):** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.
- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Develop Assembly Language programs involving Machine Control and Interrupt Instructions**’:

1. Programming skills.
2. Debugging skills.
3. Hardware Understanding

D. Relevant Course Outcomes (COs):

1. Perform Assembly language programming using 8085 Instruction Set.

E. Practical Outcomes:

1. Develop assembly language programs involving Input-Output Instructions in 8085 microprocessors.

F. Relevant Affective domain Outcomes (ADOs):

1. Appreciation for Low-level Programming.
2. Curiosity and Interest in Assembly Language.
3. Confidence in Programming.
4. Critical Thinking and Troubleshooting Skills.
5. Follow ethical practices

G. Prerequisite Theory:

Instruction	Usage	Register Transfer Language	Brief Description	Instruction Size	Addressing Mode
Machine Control Instructions (NOP, HLT)					
NOP	NOP	-	No operation. Consumes one machine cycle without performing any operation.	1 byte	Implied
HLT	HLT	-	Halt the microprocessor's execution, stopping further operations.	1 byte	Implied
Interrupt Instructions - Restart (RST 0 - 7)					
RST	RST n (0-7)	$SP \leftarrow SP - 1$; $(SP) \leftarrow PC$; $PC \leftarrow n * 8$	Call a predefined subroutine at memory location 0000h.	1 byte	Implied

Read & Set Interrupt Mask Instructions (RIM, SIM)										
RIM	RIM	Read accumulator (A) with interrupt mask status	Read the status of the interrupt enable/disable flag (IE) and store it in the accumulator.						1 byte	Implied
			SID	I 7.5	I 6.5	I 5.5	IE	M 7.5		
SIM	SIM	Interrupt Enable Flag (IE) ← data	Set the interrupt enable flag (IE) to either 0 or 1 based on the data in the accumulator.						1 byte	Implied
			SO D	SD E	X	R 7.5	MS E	M 7.5		

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: Dual Core RAM: 4 GB Operating System: Windows 7
2.	GNU 8085 Simulator Software	

I. Source code and Output:

Write an assembly language program to execute a delay routine stored at location 0010h.
Load the counter for the delay from 2050h.

Before Execution:

[2050h]: 32h

Delay Routine:

DELAY: NOP

DCR C

JNZ DELAY

Source Code:

Write an assembly language program Enable Interrupts and Clear all pending requests of RST 7.5 as well as Mask RST 7.5 and RST 5.5.

Write the Bit Pattern Here:

[7]: __, [6]: __, [5]: __, [4]: __, [3]: __, [2]: __, [1]: __, [0]: __

Source Code:

Write an assembly language program to read the interrupt mask and interpret the answer.

Source Code:

RIM

HLT

After Execution:

[A]: 9Ch (1 0 0 1 1 1 0 0)

Interpret the above output bit by bit and write your answer below:

J. Practical related Quiz:

1. Which instruction is used to perform no operation (NOP) in the 8085 microprocessor?
 - a. ADD
 - b. SUB
 - c. NOP
 - d. HLT
2. What is the purpose of the HLT instruction in the 8085 microprocessor?
 - a) Increment the accumulator value
 - b) Halt the processor and stop further execution
 - c) Load a data byte into the accumulator
 - d) Restart the microprocessor
3. Which RST instruction is used to call a predefined subroutine at memory location 0000h?
 - a) RST 0
 - b) RST 1
 - c) RST 5
 - d) RST 7
4. What does the RIM instruction do in the 8085 microprocessor?
 - a) Read a data byte from the memory
 - b) Reset the accumulator value to zero
 - c) Read the status of the interrupt enable flag (IE) into the accumulator
 - d) Set the interrupt enable flag (IE) to 1
5. The SIM instruction in the 8085 microprocessor is used to:
 - a) Simulate interrupt-driven operations
 - b) Store data from the accumulator to the memory
 - c) Enable the serial input/output mode
 - d) Set the interrupt enable flag (IE) using data from the accumulator
6. The purpose of the HLT instruction is to:
 - a) Halt the processor and stop all operations
 - b) Halt the processor for a specific time delay
 - c) Halt the processor until an interrupt occurs
 - d) Halt the processor and disable further interrupts

7. Which of the following instructions can generate a software interrupt in the 8085 microprocessors?
 - a) NOP
 - b) HLT
 - c) RST
 - d) IN
8. What is the size of the RST instruction in the 8085 microprocessor?
 - a) 1 byte
 - b) 2 bytes
 - c) 3 bytes
 - d) 4 bytes
9. Which interrupt instruction is used to call a predefined subroutine at memory location 0038h?
 - a) RST 0
 - b) RST 1
 - c) RST 5
 - d) RST 7
10. The purpose of the RIM instruction is to:
 - a) Reset the accumulator value to zero
 - b) Read the status of the interrupt mask from memory
 - c) Enable the interrupt mask
 - d) Set the interrupt enable flag (IE) to 1

Answers:

- c) NOP
- b) Halt the processor and stop further execution
- a) RST 0
- c) Read the status of the interrupt enable flag (IE) into the accumulator
- d) Set the interrupt enable flag (IE) using data from the accumulator
- a) Halt the processor and stop all operations
- c) RST
- a) 1 byte
- d) RST 7
- b) Read the status of the interrupt mask from memory

Signature

Practical No. 08: Branching & Looping Instructions

1. List Branching and Looping instructions of 8085 with example and execute basic 2-3 programs associated with said concept.

A. Objectives:

Understanding Control Flow: Introduce students to branching and looping instructions in assembly language, enabling them to understand how these instructions alter the control flow of the program.

Conditional Execution: Teach students how to use conditional branching instructions like JMP, JZ, JNZ, JC, etc., to create decision-making structures and execute specific code blocks based on certain conditions.

Loop Implementation: Provide hands-on experience in implementing loops using instructions like DJNZ, LOOP, etc., to repeat a set of instructions multiple times, which is crucial for executing repetitive tasks efficiently.

Enhanced Program Logic: Enable students to enhance program logic using branching and looping constructs, making the programs more versatile and capable of handling diverse scenarios.

Optimizing Code: Illustrate how to optimize code by using loops to reduce repetitive instructions, leading to efficient memory utilization and faster program execution.

Understanding Assembly Language Constructs: Familiarize students with different assembly language constructs like labels, jump instructions, and conditional flags, emphasizing their role in creating structured programs.

Developing Efficient Code: Encourage students to design programs that minimize resource usage, such as minimizing the number of instructions, avoiding unnecessary branches, and ensuring proper loop termination.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Engineering Tools, Experimentation and Testing (PO4):** Apply modern *engineering* tools and appropriate technique to conduct standard tests and measurements.
- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Develop Assembly Language programs involving Branching & Looping Instructions**’:

1. Programming skills.
2. Debugging skills.
3. Hardware Understanding

D. Relevant Course Outcomes (COs):

1. Perform Assembly language programming using 8085 Instruction Set.

E. Practical Outcomes:

1. Develop assembly language programs involving Input-Output Instructions in 8085 microprocessors.

F. Relevant Affective domain Outcomes (ADOs):

1. Appreciation for Low-level Programming.
2. Curiosity and Interest in Assembly Language.
3. Confidence in Programming.
4. Critical Thinking and Troubleshooting Skills.
5. Follow ethical practices

G. Prerequisite Theory:

Instruction	Usage	Register Transfer Language	Brief Description	Instruction Size	Addressing Mode
Jump Instructions					
JMP	JMP address	PC ← address	Unconditional jump to the specified memory address	3 bytes	Direct
JNZ	JNZ address	if (Z = 0) then PC ← address	Jump to the specified memory address if zero flag (Z) is not set	3 bytes	Direct
JZ	JZ address	if (Z = 1) then PC ← address	Jump to the specified memory address if zero flag (Z) is set	3 bytes	Direct
JNC	JNC address	if (CY = 0) then PC ← address	Jump to the specified memory address if carry flag (CY) is not set	3 bytes	Direct

JC	JC address	if (CY = 1) then PC ← address	Jump to the specified memory address if carry flag (CY) is set	3 bytes	Direct
JPO	JPO address	if (P = 0) then PC ← address	Jump to the specified memory address if parity flag (P) is odd (0)	3 bytes	Direct
JPE	JPE address	if (P = 1) then PC ← address	Jump to the specified memory address if parity flag (P) is even (1)	3 bytes	Direct
JP	JP address	if (S = 0) then PC ← address	Jump to the specified memory address if sign flag (S) is positive (0)	3 bytes	Direct
JM	JM address	if (S = 1) then PC ← address	Jump to the specified memory address if sign flag (S) is negative (1)	3 bytes	Direct
Call Instructions					
CALL	CALL address	SP ← (SP - 1), (SP) ← (PC + 3), PC ← address	Call the subroutine at the specified memory address	3 bytes	Direct
CNZ	CNZ address	if (Z = 0) then CALL address	Call the subroutine at the specified memory address if zero flag (Z) is not set	3 bytes	Direct
CZ	CZ address	if (Z = 1) then CALL address	Call the subroutine at the specified memory address if zero flag (Z) is set	3 bytes	Direct
CNC	CNC address	if (CY = 0) then CALL address	Call the subroutine at the specified memory address if carry flag (CY) is not set	3 bytes	Direct
CC	CC address	if (CY = 1) then CALL address	Call the subroutine at the specified memory address if carry flag (CY) is set	3 bytes	Direct
CPO	CPO address	if (P = 0) then CALL address	Call the subroutine at the specified memory address if parity flag (P) is odd (0)	3 bytes	Direct

CPE	CPE address	if (P = 1) then CALL address	Call the subroutine at the specified memory address if parity flag (P) is even (1)	3 bytes	Direct
CP	CP address	if (S = 0) then CALL address	Call the subroutine at the specified memory address if sign flag (S) is positive (0)	3 bytes	Direct
CM	CM address	if (S = 1) then CALL address	Call the subroutine at the specified memory address if sign flag (S) is negative (1)	3 bytes	Direct
Return Instructions					
RET	RET	PC \leftarrow (SP), SP \leftarrow (SP + 1)	Return from the subroutine to the calling program	1 byte	Direct
RNZ	RNZ	if (Z = 0) then RET	Return from the subroutine to the calling program if zero flag (Z) is not set	1 byte	Direct
RZ	RZ	if (Z = 1) then RET	Return from the subroutine to the calling program if zero flag (Z) is set	1 byte	Direct
RNC	RNC	if (CY = 0) then RET	Return from the subroutine to the calling program if carry flag (CY) is not set	1 byte	Direct
RC	RC	if (CY = 1) then RET	Return from the subroutine to the calling program if carry flag (CY) is set	1 byte	Direct
RPO	RPO	if (P = 0) then RET	Return from the subroutine to the calling program if parity flag (P) is odd (0)	1 byte	Direct
RPE	RPE	if (P = 1) then RET	Return from the subroutine to the calling program if parity flag (P) is even (1)	1 byte	Direct
RP	RP	if (S = 0) then RET	Return from the subroutine to the calling program if sign flag (S) is positive (0)	1 byte	Direct
RM	RM	if (S = 1) then RET	Return from the subroutine to the calling program if sign flag (S) is negative (1)	1 byte	Direct

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: Dual Core RAM: 4 GB Operating System: Windows 7
2.	GNU 8085 Simulator Software	

I. Source code and Output:

Write an assembly language program that takes an integer 'N' as input from memory location 2050h and then reads 'N' numbers from memory locations 2051h to (2050+N)h. The program should compute the sum of these 'N' numbers and store the result in memory location 2060h.

Before Execution:

[2050h]: 07h
 [2051h]: ____
 [2052h]: ____
 [2053h]: ____
 [2054h]: ____
 [2055h]: ____
 [2056h]: ____
 [2057h]: ____

Source Code:

After Execution:

[2060h]: ____

[2061h]: ____ (Carry, If Any)

Develop an assembly language program that takes an integer 'N' as input from memory location 2050h and calculates its factorial. The result should be stored in memory location 2060h.

Before Execution:

[2050h]: 04h

Source Code:

After Execution:

[2060h]: ____

Write an assembly language program that takes an 8-bit binary number as input from memory location 2050h and converts it into its decimal equivalent. The decimal value should be stored in memory location 2060h.

Before Execution:

[2050h]: 3Eh

Source Code:

After Execution:

[2060h]: ____

Write an assembly language program that reads an array of 'N' numbers from memory locations 2050h to (2050+N-1)h. The program should classify each number as even or odd and store the count of even numbers in memory location 2060h and the count of odd numbers in memory location 2061h.

Before Execution:

[2050h]: 07h

[2051h]: ____

[2052h]: ____

[2053h]: ____

[2054h]: ____

[2055h]: ____

[2056h]: ____

[2057h]: ____

Source Code:

After Execution:

[2060h]: ____

[2061h]: ____

Write an assembly language program that takes an array of 'N' numbers as input from memory locations 2050h to (2050+N-1)h. The program should find the largest number in the array and store it in memory location 2060h.

Before Execution:

[2050h]: 07h

[2051h]: ____

[2052h]: ____

[2053h]: ____

[2054h]: ____

[2055h]: ____

[2056h]: ____

[2057h]: ____

Source Code:

After Execution:

[2060h]: ____

Write an assembly language program to multiply numbers stored in memory location 2050h and 2051h. Also take care of the overflow if there's any while multiplication.

Before Execution:

[2050h]: 35h

[2051h]: 08h

Source Code:

After Execution:

[2060h]: ____

[2061h]: ____

J. Practical related Quiz:

1. Which 8085 instruction is used for an unconditional jump?
 - a) JMP
 - b) JNZ
 - c) JNC
 - d) JPE
2. Which flag is checked by the JNC instruction to determine whether the jump will be executed?
 - a) Zero flag (Z)
 - b) Carry flag (CY)
 - c) Sign flag (S)
 - d) Parity flag (P)
3. What is the purpose of the LOOP instruction in 8085 assembly language?
 - a) To execute a loop a specified number of times
 - b) To jump to a specified address unconditionally
 - c) To call a subroutine
 - d) To return from a subroutine

4. The JZ instruction in 8085 is used to jump if:
 - a) The zero flag is set
 - b) The carry flag is set
 - c) The parity flag is set
 - d) The sign flag is set
5. Which flag is checked by the JNC instruction to determine whether the jump will be executed?
 - a) Zero flag (Z)
 - b) Carry flag (CY)
 - c) Sign flag (S)
 - d) Parity flag (P)
6. What does the RST instruction in 8085 do?
 - a) Restart the microprocessor
 - b) Return from a subroutine
 - c) Read and Set the Trap flag
 - d) Return from an interrupt routine
7. Which instruction is used to implement a loop in assembly language programming?
 - a) JMP
 - b) JZ
 - c) CALL
 - d) LOOP
8. The JPO instruction in 8085 is used to jump if:
 - a) The parity flag is set
 - b) The zero flag is set
 - c) The carry flag is set
 - d) The sign flag is set
9. How many RST instructions are available in the 8085-instruction set?
 - a) 4
 - b) 5
 - c) 6
 - d) 7

10. What is the function of the CALL instruction in 8085 assembly language?
 - a) Call a subroutine
 - b) Return from a subroutine
 - c) Jump to a specified address
 - d) Restart the microprocessor
11. The JC instruction in 8085 is used to jump if:
 - a) The zero flag is set
 - b) The carry flag is set
 - c) The parity flag is set
 - d) The sign flag is set
12. The RZ instruction in 8085 is used to:
 - a) Return if Zero
 - b) Return if Not Zero
 - c) Return from Zero
 - d) Return from Not Zero
13. Which flag is checked by the JPE instruction to determine whether the jump will be executed?
 - a) Zero flag (Z)
 - b) Carry flag (CY)
 - c) Sign flag (S)
 - d) Parity flag (P)
14. What is the purpose of the SIM instruction in 8085 assembly language?
 - a) Set the interrupt mask
 - b) Reset the interrupt mask
 - c) Save the interrupt mask
 - d) Load the interrupt mask
15. The instruction JC is used to:
 - a) Jump if the carry flag is not set
 - b) Jump if the carry flag is set
 - c) Jump if the zero flag is not set
 - d) Jump if the zero flag is set

Answers:

- a) JMP
- b) Carry flag (CY)
- a) To execute a loop a specified number of times
- a) The zero flag is set
- b) Carry flag (CY)
- a) Restart the microprocessor
- d) LOOP
- a) The parity flag is set
- d) 7
- a) Call a subroutine
- b) The carry flag is set
- a) Return if Zero
- d) Parity flag (P)
- a) Set the interrupt mask
- b) Jump if the carry flag is set

Signature

Practical No. 09: Memory Hierarchy

1. Make a chart to represent all types of memory in Memory Hierarchy

A. Objectives:

Classification of various types of memories based on their characteristic and working.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Select proper memories based on requirement.**’

1. Memory concepts and its hierarchy.
2. Various types of memories working and construction principles.
3. Selection of proper memory based on requirement.

D. Relevant Course Outcomes (COs):

1. Characterize need of various Memory types in hierarchy

E. Practical Outcomes: Apply proper memory based on requirement.

1. Memory classifications and Memory Hierarchy
2. Main Memory and RAM, ROM, PROM, EPROM
3. Auxiliary Memory
4. Associative Memory
5. Cache Memory
6. Virtual memory

F. Relevant Affective domain Outcomes (ADOs):

4. Classification of memory based on construction and working.
5. Selection of memory based on application.

G. Prerequisite Theory:

Hardware fundamentals of storage element, bit, Nibble, Byte, word etc.

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Chart, Display Board, Model or module of various types of memories for Learners to get familiar with various types of memories their construction and working.	

I. Learner Centric Activities and Outcome:

- Collect various types of Discs and make a Chart with Explanation
- Prepare chart of memory hierarchy
- Make a small poster to represent all types of memory in Memory Hierarchy.
- Compare all types of Memory in details etc.

Signature

Practical No. 10: Associative Memory

1. Paraphrase Associative Memory in detail.

A. Objectives:

Understanding Associative Memory: Gain an understanding of associative memory, a type of computer memory that allows the system to perform high-speed searches based on content rather than addresses.

Familiarity with Content Addressable Memory (CAM): Explore the concept of Content Addressable Memory (CAM), which enables retrieval of data based on its content rather than its physical address.

Working Principle of Associative Memory: Learn about the working principle of paraphrasing associative memory, where a partial match of data with its paraphrased version is considered a successful search.

Understanding Associative Memory Architecture: Study the architecture of associative memory, including word lines, bit lines, match lines, and parallel comparison of data.

Optimization Techniques: Investigate optimization techniques to enhance the speed and efficiency of the paraphrasing associative memory.

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Examine computer architecture**':

1. Competency in understanding microprocessor architecture & Various Memories: Students will gain a solid understanding of how different types of memory devices work and when to use which type of memory.

D. Relevant Course Outcomes (COs):

1. Characterize need of various Memory types in hierarchy.

E. Practical Outcomes:

1. **Understanding of Associative Memory Concepts:** Students will gain a clear understanding of associative memory and its functioning as a content-addressable memory, enabling high-speed data retrieval based on content matching.
2. **Knowledge of Associative Memory Architecture:** Students will learn about the architecture of associative memory, including word lines, bit lines, and match lines, as well as the mechanism of parallel data comparison.

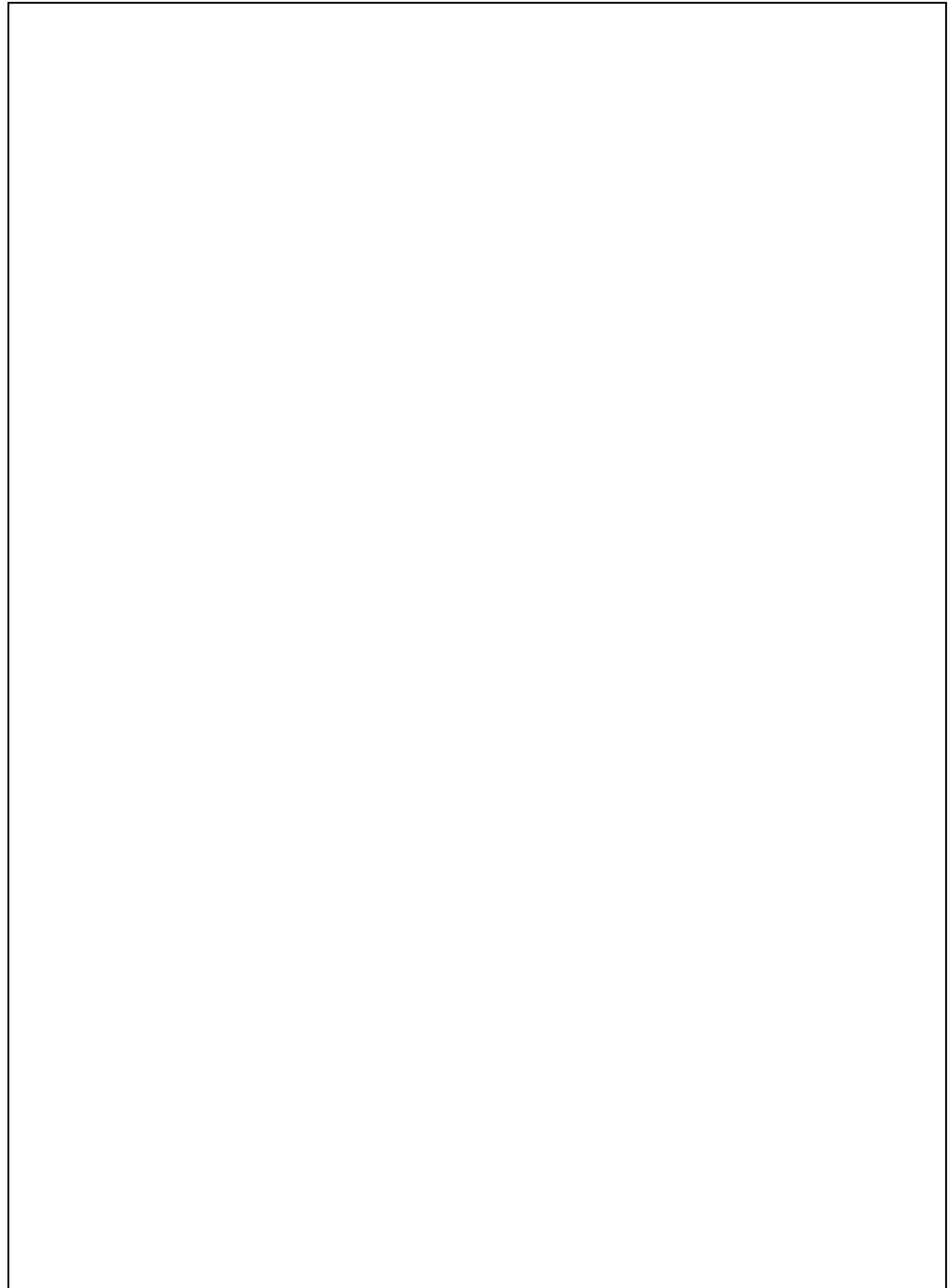
F. Relevant Affective domain Outcomes (ADOs):

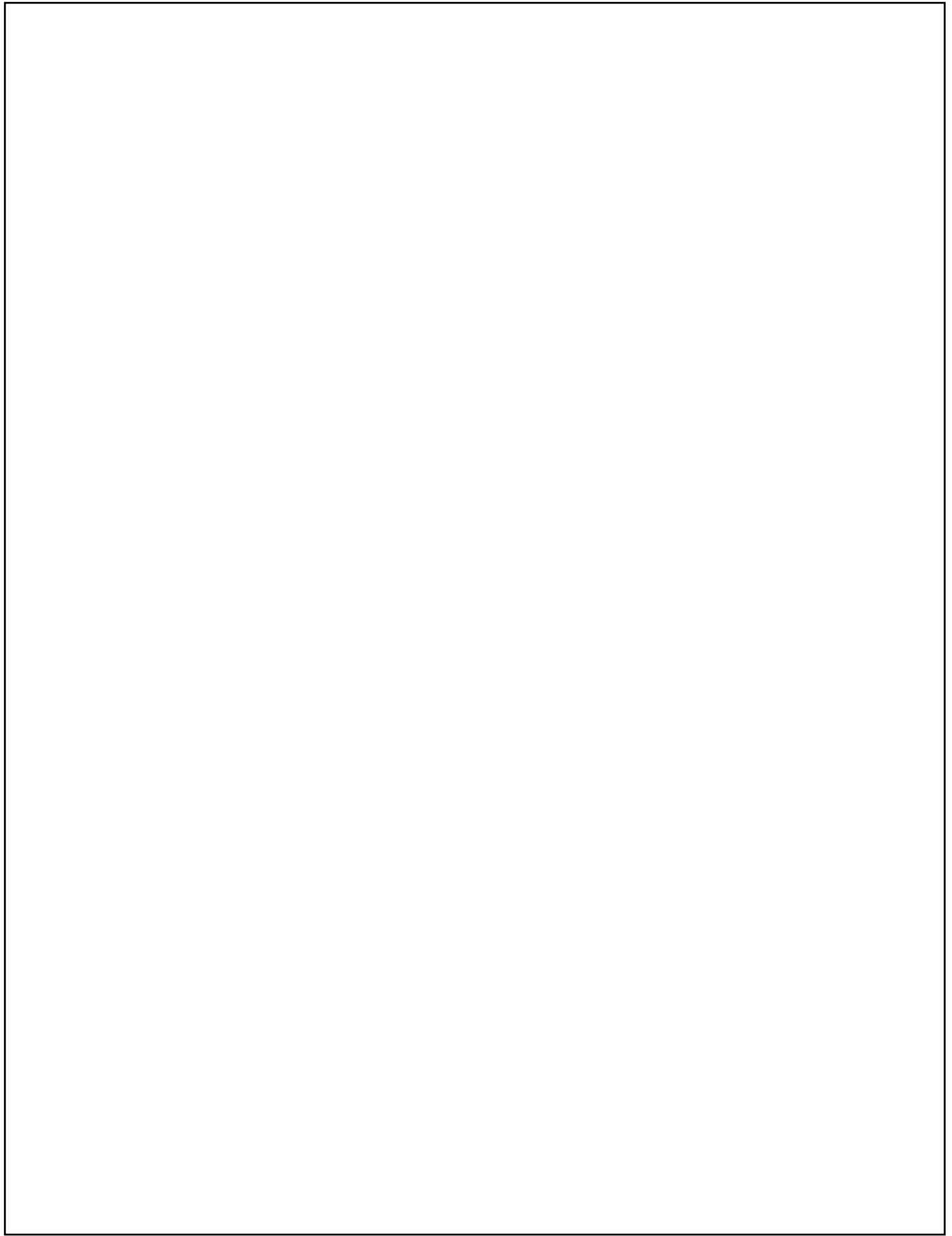
1. Enhanced Appreciation for Memory Systems
2. Curiosity and Inquisitiveness
3. Sense of Achievement
4. Passion for Technology.
5. Awareness of Real-world Relevance

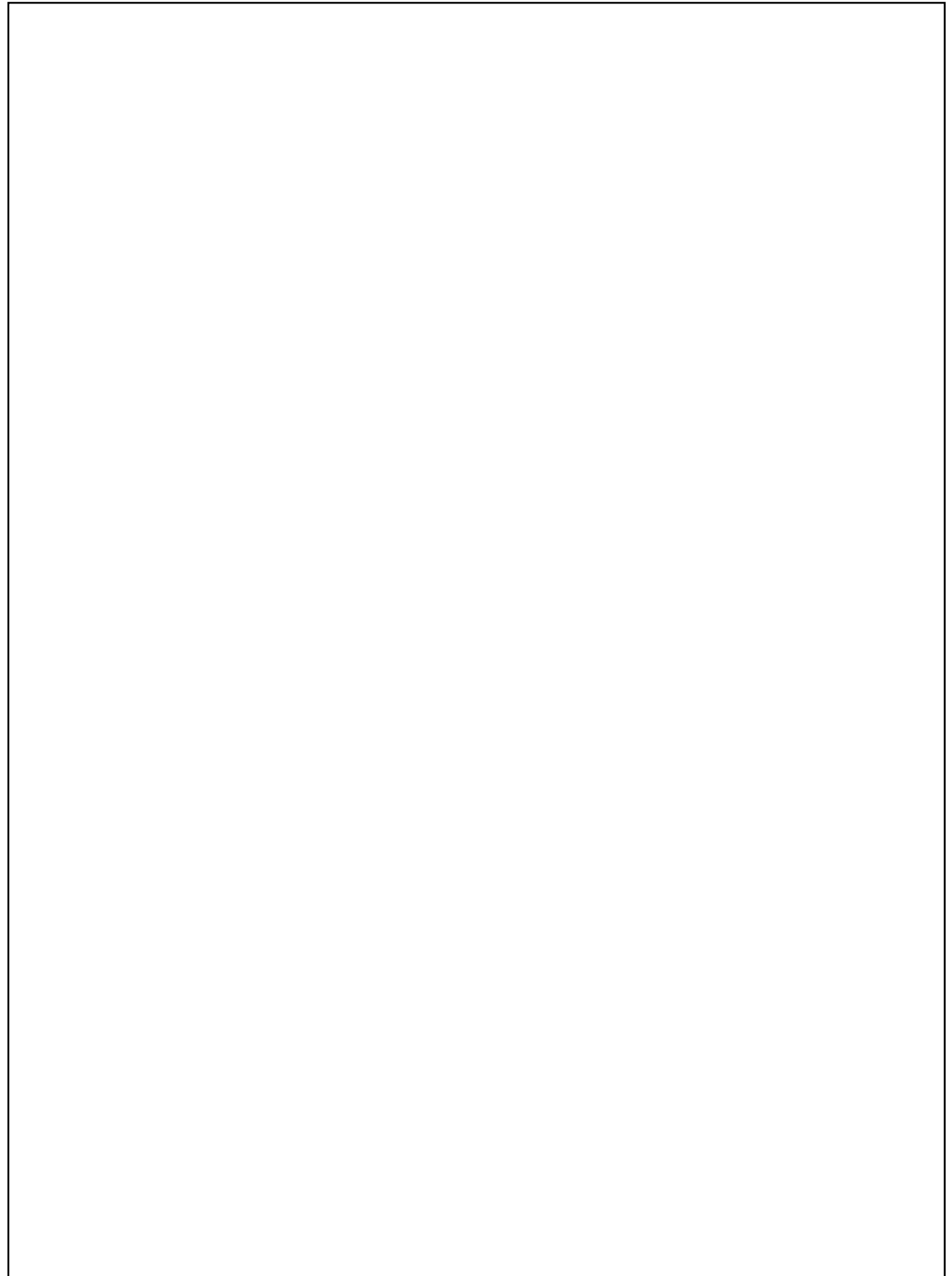
G. Prerequisite Theory:

- **Digital Logic:** Understanding digital logic gates, Boolean algebra, and basic digital Basic Computer Architecture: Students should have a solid understanding of the basic architecture of a computer, including the CPU, memory units, and data storage.
- **Binary Number System:** Proficiency in the binary number system is essential, as memory addresses and data representation in computer systems are based on binary digits.
- **Logic Gates and Boolean Algebra:** Knowledge of logic gates (AND, OR, NOT) and Boolean algebra is necessary to comprehend the fundamental building blocks of digital circuits.
- **Memory Hierarchy:** Familiarity with the memory hierarchy, including cache, primary memory (RAM), and secondary memory (hard disk, SSD), is important to grasp the role of associative memory in the memory hierarchy.
- **Data Representation:** Understanding how data is represented in different formats (e.g., binary, hexadecimal, decimal) is crucial for working with data in memory units.
- **Pattern Matching Techniques:** Basic knowledge of pattern matching techniques and algorithms can aid in understanding how associative memory performs searches based on content matching.
- **Digital Circuits and Logic Design:** A grasp of digital circuits and logic design is beneficial for comprehending the underlying hardware implementation of associative memory units.

H. Explain working of Associative Memory in brief with neat diagram.







I. Practical related Quiz:

1. What is the primary purpose of the Associative Memory in computer systems?
 - a) To store data in a sequential manner
 - b) To enable high-speed content-based data retrieval
 - c) To perform arithmetic and logical operations
 - d) To manage input and output devices
2. Which type of memory allows the system to perform high-speed searches based on content rather than addresses?
 - a) Random Access Memory (RAM)
 - b) Read-Only Memory (ROM)
 - c) Content Addressable Memory (CAM)
 - d) Cache Memory
3. In associative memory, what is considered a successful search?
 - a) Exact match of data with its stored version
 - b) Partial match of data with its paraphrased version
 - c) A match of the memory address with the data content
 - d) Retrieval of data based on the least significant bit (LSB)
4. Which of the following is a fundamental building block of associative memory used for parallel comparison of data?
 - a) Word line
 - b) Bit line
 - c) Match line
 - d) Clock signal
5. What advantage does associative memory offer over traditional memory systems like RAM?
 - a) Lower cost and higher storage capacity
 - b) Faster access and retrieval speed based on data content
 - c) More reliable and durable storage of data
 - d) Reduced power consumption and heat generation

Answers:

- b) To enable high-speed content-based data retrieval
- c) Content Addressable Memory (CAM)
- a) Exact match of data with its stored version
- c) Match line
- b) Faster access and retrieval speed based on data content

Signature

Practical No. 11: Input-Output

1. Differentiate Programmed I/O and Interrupt initiated I/O in detail.

A. Objectives:

Visualize CPU-I/O Communication

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Visualize CPU-I/O Communication.**’

1. In depth knowledge of Programmed I/O and Interrupt initiated I/O.
2. Comparison of Programmed I/O and Interrupt initiated I/O in detail.

D. Relevant Course Outcomes (COs):

2. Visualize CPU-I/O Communication and working.

E. Practical Outcomes: Apply proper memory based on requirement.

7. CPU internal and external communication with peripherals.

F. Relevant Affective domain Outcomes (ADOs):

1. Exposure to various I/O interfaces.
2. Execution of CPU communication with peripherals in programmed I/O mode
3. Execution of CPU communication with peripherals with Interrupt initiated I/O.

Practical No. 12: CPU-IOP Communication

1. List steps to carryout CPU-IOP Communication.

A. Objectives:

Visualize CPU-IOP Communication

B. Relevant Program Outcomes (POs):

- **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the *engineering* problems.
- **Problem analysis (PO2):** Identify and analyse well-defined *engineering* problems using codified standard methods.
- **Design/development of solutions (PO3):** Design solutions for *engineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
- **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Visualize CPU-IOP Communication.**’

1. Validation of CPU-IOP Communication.

D. Relevant Course Outcomes (COs):

3. Visualize CPU-I/O Communication and working.

E. Practical Outcomes: Apply proper memory based on requirement.

8. CPU internal and external communication with peripherals.

F. Relevant Affective domain Outcomes (ADOs):

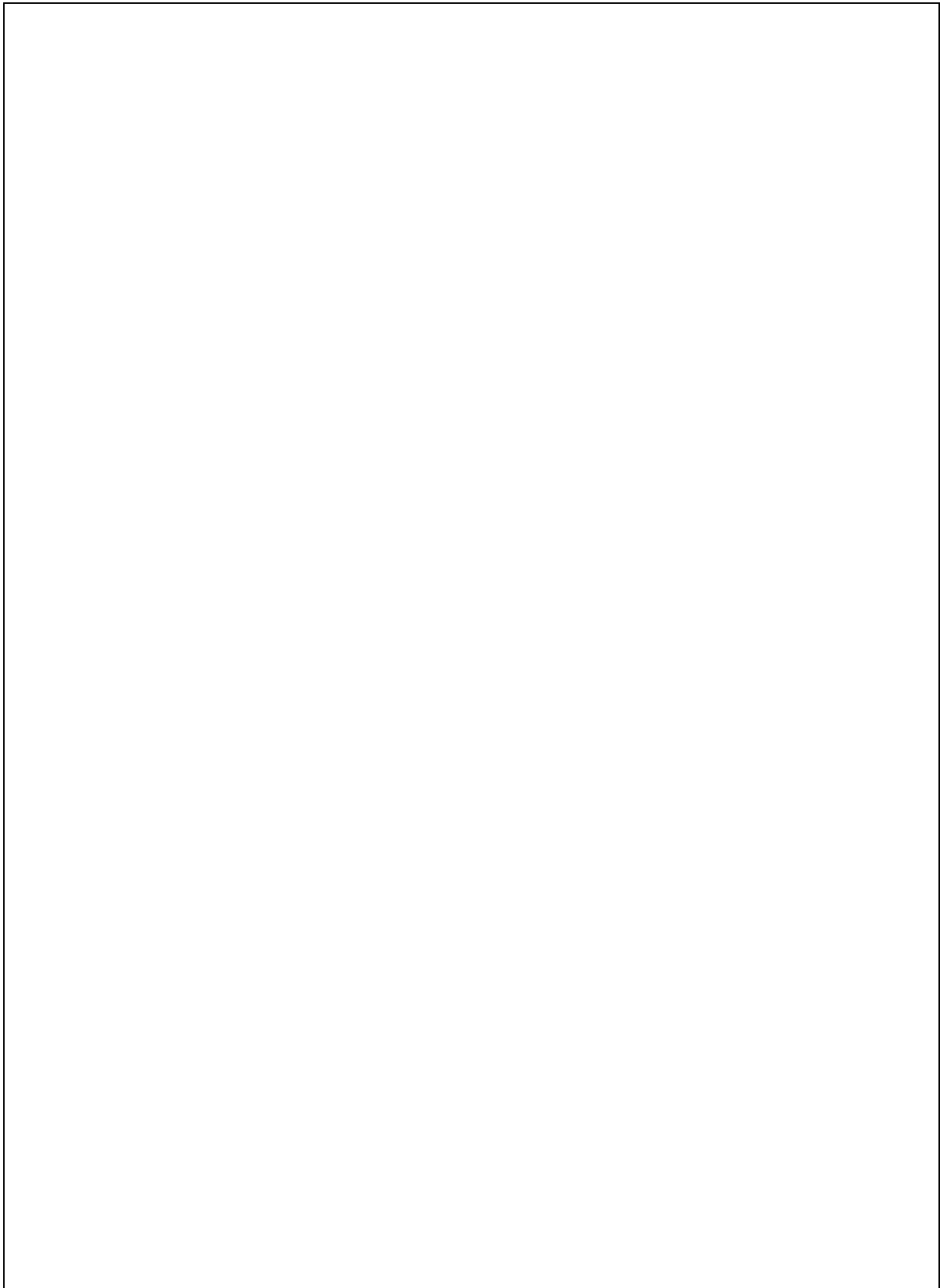
1. Exposure to need of IOP.
2. Explain CPU-IOP Communication.

G. Prerequisite Theory:

Hardware fundamentals related to various I/O interface and data communication fundamentals are Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Validate CPU-I/O Communication and working in programmed and interrupt driven mode.	

H. Draw sequence of CPU-IOP Communication:



Signature