IMPLEMENTATION OF BINARY SEARCH USING 8085 ASSEMBLY LANGUAGE

PROJECT REPORT

21CSS201T - COMPUTER ORGANIZATION AND ARCHITECTURE

(2021 Regulation)

II Year / III Semester

Academic Year: 2023 -2024

By

NAGA SUDHEER CH (RA2211026010558)

K RUFAS (RA2211026010557)

B PAVAN (RA2211026010540)

Under the guidance of

Dr. K Suresh Assistant Professor Department of Computational Intelligence

FACULTY OF ENGINEERING AND TECHNOLOGY SCHOOL OF COMPUTING SRM INSTITUTE OF SCIENCE AND TECHNOLOGY Kattankulathur, Kancheepuram

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE

This is to certify that **21CSS201T COMPUTER ORGANIZATION AND ARCHITECTURE** project report titled "**IMPLEMENTATION OF BINARY SEARCH USING 8085 ASSEMBLY**" is the bonafide work of NAGA SUDHEER CH **(RA2211026010558), K RUFAS (RA2211026010557), AND B PAVAN (RA2211026010540)** who undertook the task of completing the project within the allotted time.

SIGNATURE	**SIGNATURE**
Dr. K Suresh	Dr. Annie Uthra
Assistant Professor	**Professor and Head**
Department of CINTEL,	Department of CINTEL
SRM Institute of Science and Technology	SRM Institute of Science and Technology

About the course:-

21CSS201T - COMPUTER ORGANIZATION AND ARCHITECTURE is three credit course with **L T P C as 3-0-0-3**

Objectives:

The purpose of learning this course is to

Course Learning Rationale (CLR): The purpose of learning this course is to:

Course Learning Rationale (CLR):	The purpose of learning this course is to:
CLR-1	Understand the Fundamentals of computers, Memory operations and Addressing Modes
CLR-2	Know about Functions of Arithmetic and Logic unit
CLR-3	Explore the Operations of Control Unit, Execution of Instruction and Pipelining
CLR-4	Classify the Need for Parallelism, Multicore and Multiprocessor Systems
CLR-5	Understand the Concepts and functions of Memory unit, I/O unit

Course Learning Outcomes (CLO): At the end of this course, learners will be able to:

Course Learning Outcomes (CLO):	At the end of this course, learners will be able to:
CO-1	Identify the computer hardware and how software interacts with computer hardware
CO-2	Apply Boolean algebra as related to designing computer logic ,through simple combinational and sequential logic circuits
CO-3	Examine the detailed operation of Basic Processing units and the performance of Pipelining
CO-4	Analyze concepts of parallelism and multi-core processors.
CO-5	Classify the memory technologies, input-output systems and evaluate the performance of memory system

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE

DEPARTMENT OF
SCHOOL OF COMPUTING
College of Engineering and Technology
SRM Institute of Science and Technology

MINI PROJECT REPORT

ODD Semester, 2023-2024

Lab code & Sub Name	:21CSS201T & Computer Organization and Architecture
Year & Semester	:II & III
Project Title	: Implementation of binary search using 8085 assembly language
Lab Supervisor	: **Mr.K.SURESH**
Team Members	: 1. SUDHEER CH (Reg.No:RA2211026010558) 2.RUFAS K(Reg.No: RA2211026010557) 3.PAVAN B(Reg.No: RA2211026010540)

Particulars	**Max. Marks**	**Marks Obtained**
		Name:
		Register No :
Program and Execution	20	
Demo verification &viva	15	
Project Report	05	
Total	**40**	

Date :

Staff Name : **Mr.K.SURESH**

Signature :

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy,** for his encouragement.

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal,** for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman,** for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professors Dr. Vadivu. G , Professor, Department of Data Science and Business Systems and Dr. Sasikala. E Professor, Department of Data Science and Business Systems** and **Course Coordinators** for their constant encouragement and support.

We are highly thankful to our my Course project Faculty **Mr. K suresh, Assistant professor , CINTEL,** for her assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **Annie Uthra, Head of Computer Intelligence, CINTEL** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course

IMPLEMENTATION OF BINARY SEARCH USING 8085 ASSEMBLY LANGUAGE

OBJECTIVE:

The objective of implementing the binary search algorithm in 8085 assembly language is to demonstrate the ability to efCiciently perform a fundamental searching operation on a sorted dataset using a limited-resource microprocessor

ABSTRACT:

This project focuses on the implementation of the binary search algorithm using 8085 assembly language. The 8085 microprocessor is a widely used microprocessor that offers a robust instruction set for performing various computational tasks. The binary search algorithm is a fundamental algorithm in computer science used to efCiciently locate a target element within a sorted dataset.
The project begins with a detailed introduction to the 8085 microprocessor architecture, highlighting key components such as registers, Clags, memory, and instruction set. The binary search algorithm is explained in depth, emphasizing its efCiciency and the underlying principles of divide-and-conquer.

INTRODUCTION:

The implementation of the binary search algorithm in 8085 assembly language is a fascinating endeavor that combines the principles of computer architecture and algorithmic efficiency. The 8085 microprocessor, renowned for its simplicity and versatility, provides a compelling platform to showcase how fundamental algorithms can be executed in constrained computing environments.

Binary search is a cornerstone algorithm in computer science, celebrated for its efficiency in locating a target element within a sorted dataset. By iteratively halving the search space, binary search dramatically reduces the number of comparisons required compared to linear search. This makes it an invaluable tool for quickly finding elements in large, ordered collections.

The 8085 microprocessor, though modest in computational power compared to modern processors, offers a rich set of instructions and registers. Its architecture includes a set of general-purpose registers, flags to indicate various conditions, and an array of instructions for arithmetic, logical, and control operations. This project leverages these capabilities to implement the binary search algorithm effectively.

In this project, we embark on a journey to bridge the gap between algorithmic theory and practical execution. We'll delve into the intricacies of the 8085 microprocessor, gaining an understanding of its architecture and instruction set. This knowledge will serve as the foundation for crafting an efficient binary search routine tailored to the microprocessor's capabilities.

The project not only aims to yield a functional implementation but also emphasizes robust input handling, edge case management, and performance evaluation. We'll explore how to gracefully handle scenarios where the target element is not found in the dataset and ensure the program executes optimally within the constraints of the 8085 microprocessor.

Through this endeavor, we aim to highlight the adaptability and versatility of the 8085 microprocessor and showcase the power of algorithmic thinking in solving real-world problems even within resource-constrained environments. This project serves as a testament to the enduring relevance of both classic algorithms and foundational computer architectures in the ever-evolving landscape of computing.

HARDWARE/SOFTWARE REQUIREMENTS:

Hardware Requirements:

8085 Microprocessor: This is the central processing unit where the program will be executed. It includes the ALU (Arithmetic Logic Unit), registers, and control unit.
Memory Unit: This includes both ROM (Read-Only Memory) for storing the program and RAM (Random Access Memory) for storing data during execution.
Input/Output Devices: These are used for user interaction. For this project, a simple input device like a keyboard and an output device like a display or LEDs might be sufficient.
Buses and Control Lines: These are necessary to connect various components of the microprocessor, allowing them to communicate.
Clock Source: A clock generator or crystal oscillator is required to provide clock pulses to the microprocessor, synchronizing its operations.
Power Supply: A stable power source is crucial for the proper functioning of the microprocessor and associated components.

Software Requirements:

Assembler: A software tool that translates assembly language code into machine code (binary) that the 8085 microprocessor can execute. Examples include assemblers like ASEM-85, ASMacro, etc.
Simulator or Emulator: To test the program without the actual hardware, you can use an 8085 simulator or emulator. This allows you to run and debug your program on a computer.
Text Editor: A simple text editor is needed to write and edit the assembly language code.
Hexadecimal Editor (Optional): This tool allows you to view and edit the machine code in hexadecimal format, which can be useful for debugging.

Hardware Interface (Optional): If you're planning to run the code on actual hardware, you might need an interface to connect your computer to the microprocessor.
Documentation Tools (Optional): For documenting your code, you might need tools like a word processor or LaTeX for creating reports or manuals.

CONCEPTS/WORKING PRINCIPLE

The implementation of binary search in 8085 assembly language leverages the principles of both the binary search algorithm and the architecture of the 8085 microprocessor.

1. Binary Search Algorithm:
   - Binary search is a divide-and-conquer algorithm used to find a target value within a sorted dataset. It works by repeatedly dividing the search space in half, narrowing down the possibilities with each iteration. This leads to a significant reduction in the number of comparisons needed compared to linear search.

2. 8085 Assembly Language:
   - The 8085 microprocessor is a simple yet powerful microprocessor with a defined set of instructions, registers, and memory. It operates on binary data and executes instructions in a sequential manner.

Working Steps:

1. Input Handling:
   - The program prompts the user to input a sorted array and the target element to be searched.

2. Initialization:
   - The program initializes the necessary registers and memory locations. This includes setting up the start and end indices of the array, initializing pointers, and storing the target element.

3. Binary Search Algorithm:
   - The algorithm is executed using a loop structure.
   - The mid-element of the current search space is calculated.
   - The mid-element is compared with the target value.
   - Based on the comparison, the search space is halved, narrowing down the possibilities.

4. Loop Iterations:
   - The binary search algorithm continues in a loop until one of the following conditions is met:
      - The target element is found.
      - The search space is reduced to zero, indicating the element is not present.

5. Output Display:
   - Upon successful execution, the program displays the index of the target element if found, or a message indicating its absence.

6. Efficiency Analysis:
   - The program's execution time is analyzed in terms of clock cycles. It is compared with the theoretical expectations based on the binary search algorithm's time complexity.

Block Diagram:

Creating a visual diagram is beyond my capabilities as a text-based AI. However, I can provide a description of the components:

1. Input Module:
   - Responsible for accepting user input for the sorted array and target element.

2. Registers and Memory:
   - Accumulator and other registers store data and intermediate results.
   - Memory stores the sorted array and other variables.

3. ALU (Arithmetic Logic Unit):
   - Performs arithmetic and logical operations.

4. Control Unit:
   - Orchestrates the flow of operations and instructions.

5. Program Counter (PC) and Instruction Register (IR):
   - PC keeps track of the current instruction being executed.
   - IR holds the current instruction.

6. Clock and Timing Unit:
   - Provides clock pulses for synchronization.

7. Control Signals:
   - Read, Write, Enable, etc., for managing data flow.

8. Output Module:
   - Displays the output (index of target element or "not found" message).

9. Loop Control:
   - Manages the iterative process of the binary search algorithm.

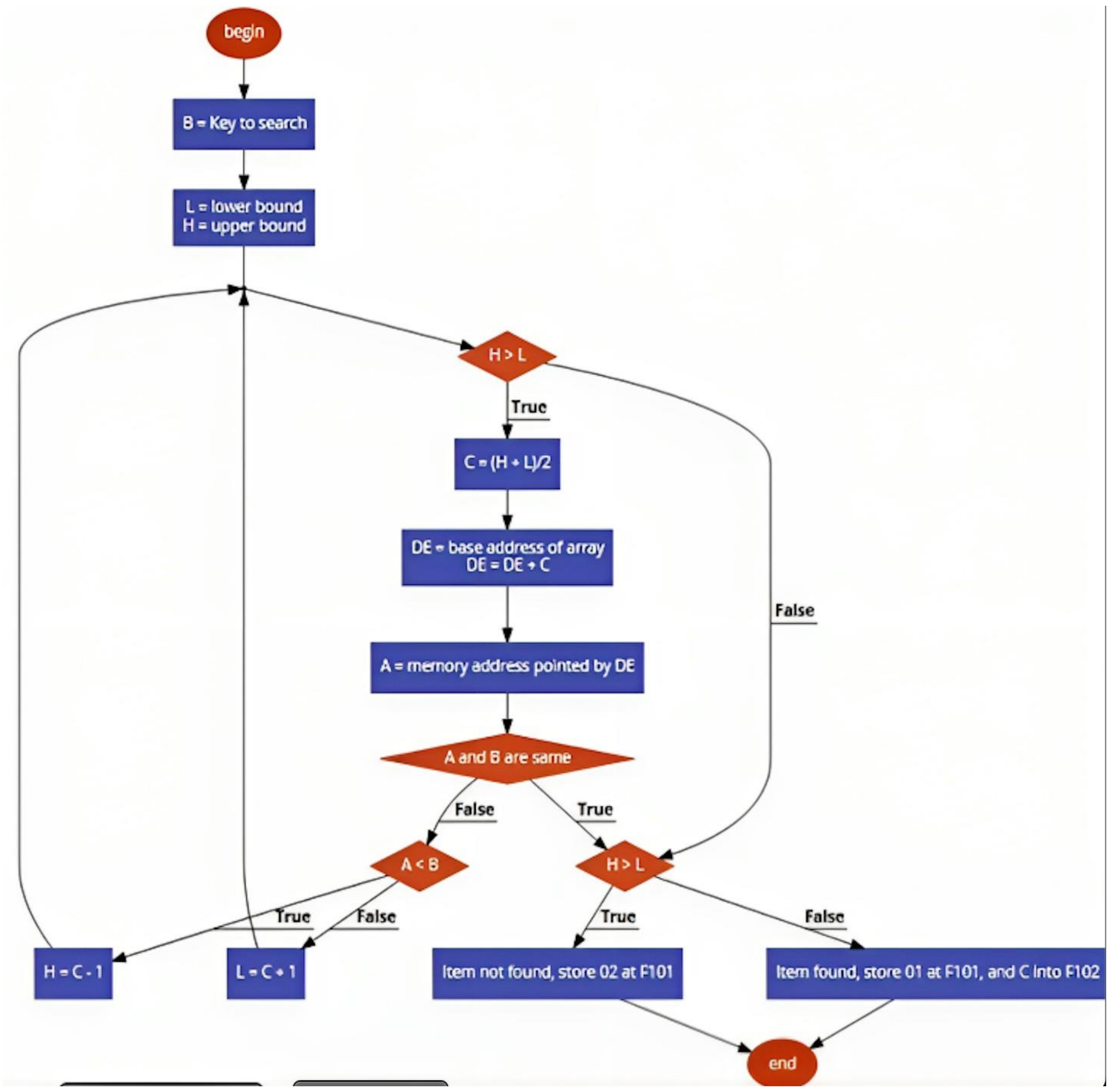
10. Efficiency Analysis Unit (Optional):
   - Measures and analyzes execution time.

APPROACH/METHODOLOGY/PROGRAMS:

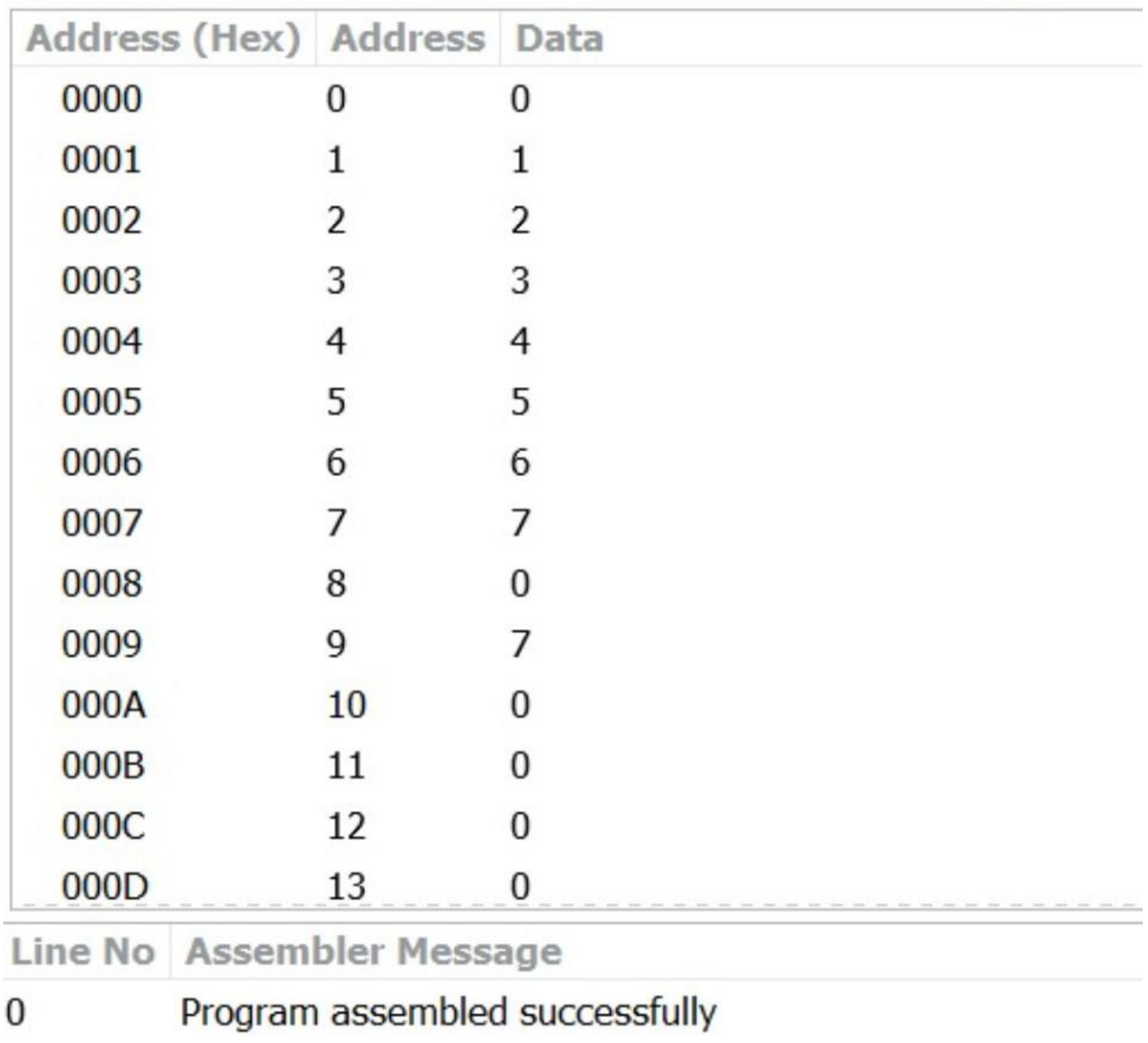
```
 mvi b, 7 ; last index to fill untill
 mvi h, 0 ; set up HL as memory pointer
mvi l, 0 ; set up HL as memory pointer
loop: inr l ; increase l by one
 mov m, l
 mov a, l
 cmp b
 jm loop
 mvi b, 0
 mvi c, 7
 mvi d, 7 ; needle@d - value to find
 mvi l, 9 ; result will be stored to memory at address 9
 mvi m, 255 ; clear result, 255 i.e. 0xFF means not found

while: mov a, c
 cmp b ; high > low
 jm exit
 ; ((low + high) >>> 1) + low;
 add c ; a already have low@b so let's add high@c
 rar ; shift to right i.e. divide by 2 so we can get an average
 add b ; average + low
 mov e, a ; middle@e
 ; get element from memory
 mov l, e ; memory address@l is middle@e
 mov a, m ; get to element@a from memory@m by index@l
 cmp d ; is element@a > needle@d?
 jz equals ; jz - jump on zero
 jm less ; jm - jump on negative
 jp greater ; jp - jump on positive
 equals: mvi l, 9 ; store middle@e to memory on 9
 mov m, e
 jmp exit
 less: mov a, e
 inr a
 mov b, a
 jmp while
 greater: mov a, e
 dcr a
 mov c, a
 jmp while
 exit: hlt
```

FLOWCHART:


begin
B = Key to search
L = lower bound
H = upper bound
H > L
True
C = (H + L)/2
DE = base address of array
DE = DE + C
A = memory address pointed by DE
A and B are same
False
True
False
A < B
H > L
True
False
True
False
H = C - 1
L = C + 1
Item not found, store 02 at F101
Item found, store 01 at F101, and C into F102
end

OUTPUT:



Address (Hex)	Address	Data
0000	0	0
0001	1	1
0002	2	2
0003	3	3
0004	4	4
0005	5	5
0006	6	6
0007	7	7
0008	8	0
0009	9	7
000A	10	0
000B	11	0
000C	12	0
000D	13	0

Line No	Assembler Message
0	Program assembled successfully

CONCLUSIONS:

Implementation of binary search using 8085 assembly language assembled successfully.