IMPLEMENTATION OF BINARY SEARCH USING 8085 MACHINE LANGUAGE

MINOR PROJECT REPORT

By

RAUNAK NAIK (RA2211003010180)
ADYA SINGH (RA2211003010181)
SHASHANK SAXENA (RA2211003010147)

Under the guidance of

Karthikeyan M

In partial fulfillment of the Course

of

21CSS201T – COMPUTER ORGANIZATION AND ARCHITECTURE

FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this minor project report for the course **21CSS201T – COMPUTER ORGANIZATION AND ARCHITECTURE** entitled "**Implementation of Binary Search Using 8085 Machine Language**" is the bonafide work of **Raunak Naik (RA2211003010180), Adya Singh (RA2211003010181) and Shashank Saxena (RA2211003010147)** carried out the work under my supervision.

SIGNATURE

Karthikeyan M
SRM Institute of Science and Technology
Kattankulathur

ABSTRACT

This project report implements Binary Search using 8085 machine language. The objective of this project is to perform Binary Search technique that works efficiently on sorted lists. We are using Sim8085, which is Sim8085 is a online development environment for writing Intel 8085 microprocessor code. It can assemble, debug 8085 assembly code and simulate the 8085 microprocessor.

The endeavor involved overcoming the inherent limitations of the 8085 microprocessor, and meticulously designing and executing an algorithm that efficiently searches for elements in a sorted dataset. Through this project, we delved into the world of low-level programming and microprocessor constraints, gaining valuable insights into the intricacies of algorithm optimization. This project not only showcases the historic significance of the 8085 microprocessor but also highlights the enduring relevance of Binary Search as a fundamental algorithm.

TABLE OF CONTENTS

1. INTRODUCTION

Binary search is the search technique that works efficiently on sorted lists. Hence, to search an element into some list using the binary search technique, we must ensure that the list is sorted.

Binary search follows the divide and conquer approach in which the list is divided into two halves, and the item is compared with the middle element of the list. If the match is found then, the location of the middle element is returned. Otherwise, we search into either of the halves depending upon the result produced through the match.

It is a fundamental algorithm with applications in various domains, including data retrieval, database management, and information retrieval systems. Implementing binary search in assembly language for the 8085 microprocessor can serve as a foundation for more complex projects and systems where resource constraints are a concern. It also provides a practical exercise in understanding the intricacies of microprocessor programming.

his project aims to harness the capabilities of the 8085 microprocessor to perform binary search operations on sorted data. In this introduction, we will provide an overview of the project, its significance, and the steps involved in the implementation.

2. LITERATURE SURVEY

Implementing binary search in assembly language for the 8085 microprocessor can be a challenging but rewarding project. Here's a literature survey to help you get started:

1. 8085 Microprocessor Documentation:
   - Begin by thoroughly understanding the 8085 microprocessor. Consult its official documentation or user manual to gain insight into its instruction set, addressing modes, and hardware architecture.

2. Assembly Language Programming:
   - To implement binary search in the 8085 assembly language, you should have a strong understanding of assembly language programming. Look for assembly language programming books or online tutorials.

3. Binary Search Algorithm:
   - Study the binary search algorithm in detail. Understand how it works, its time complexity, and the key components of the algorithm (e.g., comparing elements, dividing the array, etc.).

4. Pseudocode and High-Level Logic:
   - Before diving into assembly language, write a high-level pseudocode or algorithm for the binary search. This will help you in breaking down the problem into smaller steps.

5. Data Structures:
   - Determine the data structure (e.g., an array) in which you will perform the binary search. Understand how data is organized in memory.

6. Memory Organization:

- Study the memory organization in the 8085 microprocessor. Understand the address space, stack, and how data is stored and accessed in memory.

7. Assembly Language Programming Tools:

- Identify the tools and software you'll need for programming and testing your 8085 assembly code. You might need an assembler, simulator, and debugger.

8. Binary Search Examples:

- Look for example implementations of binary search in other assembly languages. Even though they may not be for 8085, they can give you insights into the structure and logic of the algorithm..

9. Error Handling and Testing:

- Study techniques for error handling and testing in assembly language to ensure the robustness of your binary search implementation.

3. REQUIREMENT ANALYSIS

To implement Binary Search using 8085 Machine Language, following requirements are needed -

1. 8085 Microprocessor Kit: You'll need access to an 8085 microprocessor kit or simulator. This kit includes the microprocessor chip, memory, input/output ports, and other necessary components for programming and execution.

2. Programming Software: You'll require a software tool or simulator for programming the 8085 microprocessor in machine language. Popular choices include EMU8086, Microprocessor 8085 Simulator Software, or using an online emulator.

3. Binary Search Algorithm: A clear understanding of the Binary Search algorithm is essential. You should be able to write the step-by-step instructions for this algorithm in machine language.

4. Assembly Language Knowledge: Proficiency in 8085 assembly language is required. You should be familiar with the instruction set, memory addressing, registers, and stack operations.

5. Text Editor: You'll need a text editor or integrated development environment (IDE) for writing and editing the assembly code.

6. Assembler: You'll require an assembler that can translate your assembly code into machine code that the 8085 microprocessor can execute.

7. Test Data: Prepare test data sets for verifying the correctness of your Binary Search implementation. Include both sorted and unsorted data.

8. Documentation: Document your code and project thoroughly, including comments, flowcharts, and any necessary explanations.

9. Debugging Tools: Debugging tools and techniques are crucial for identifying and resolving any issues in your code.

10. Testing Environment: Set up a controlled testing environment where you can run your code on the 8085 microprocessor kit or simulator.

4. ARCHITECTURE AND DESIGN

8085 Microprocessor Architecture:

The 8085 microprocessor has a basic architecture that includes registers, memory, and control signals :

Registers: The 8085 has various registers, including the accumulator (A), general-purpose registers (B, C, D, E, H, L), the stack pointer (SP), and the program counter (PC).

Memory: The microprocessor communicates with memory using addresses and data buses. In the case of 8085, it has a 16-bit address bus (allowing access to 64KB of memory) and an 8-bit data bus.

Control Unit: The control unit generates control signals to manage the operations of the microprocessor, including reading and writing to memory, arithmetic and logic operations, and branching.

Designing the Binary Search Algorithm:

The Binary Search algorithm is a divide-and-conquer algorithm that efficiently searches for an element in a sorted array. When implementing it on the 8085, consider the following aspects:

Data Representation: You need to store the sorted data in memory. Ensure that the data is sorted in ascending order and is accessible to the 8085's memory.

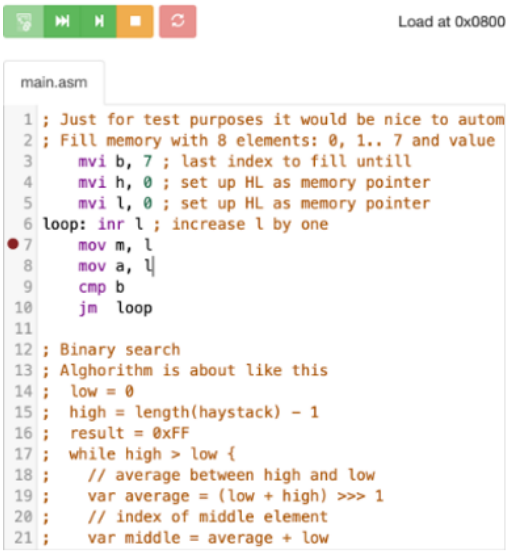
Algorithm Implementation: Design the algorithm with a focus on using 8085 assembly language instructions. The algorithm involves dividing the search interval in half, comparing values, and adjusting the search range accordingly.

Memory Management: Reserve memory locations for data storage, temporary variables, and tracking the low and high bounds of the search range.

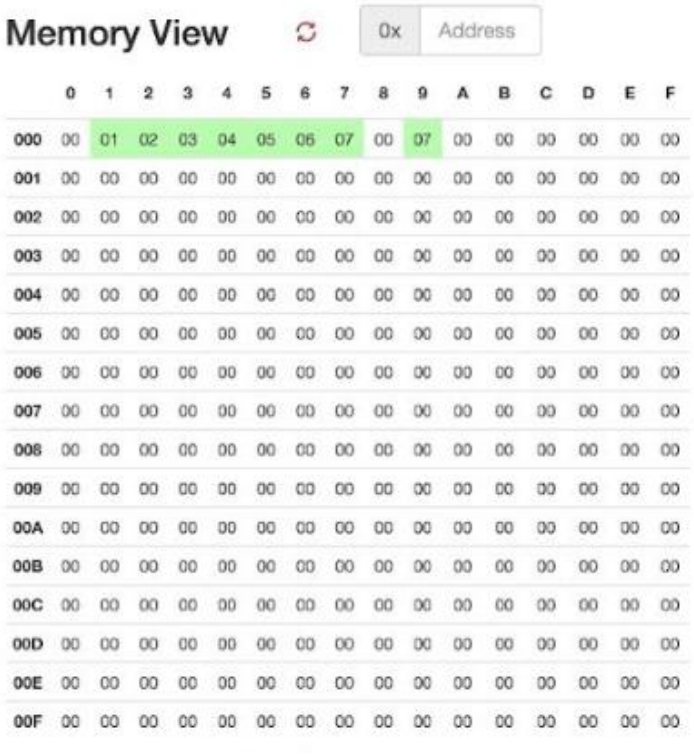
Control Flow: Implement the control flow of the algorithm using conditional branches (e.g., jump instructions) based on comparison results.

5. IMPLEMENTATION

1. Initialization: The code initializes an array (haystack) with values from 0 to 7, stored in memory locations 0 to 7.
2. Binary Search: It performs a binary search for a specific value (the "needle"), which is set to 7 in this case.
3. Result Storage: The result of the binary search, which can be the index of the found needle or 0xFF (indicating not found), is stored in memory location 9.
4. Binary Search Logic: The binary search is implemented with a loop that continues until the high index is no longer greater than the low index. In each iteration, the middle index is calculated, and the element at the middle index is compared to the needle. Based on this comparison, the low and high indices are adjusted to narrow down the search range.



6. RESULTS AND DISCUSSION



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	00	01	02	03	04	05	06	07	00	07	00	00	00	00	00	00
001	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
002	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
003	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
004	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
005	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
006	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
007	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
008	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
009	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Memory View: This typically represents the contents of the memory locations used by your 8085 program. It shows the values stored in memory addresses, often in hexadecimal format. This view helps you understand the current state of memory and can be useful for debugging and monitoring data stored in memory during program execution.

The final result, which is the index of the needle (if found) or 0xFF (not found), will be stored in memory location 9. The actual value of memory location 9 depends on the position of the needle in the haystack. If the needle is present in the haystack, the value stored in memory location 9 will be the index of the needle. If the needle is not present in the haystack, the value will remain 0xFF. The specific output will vary based on the actual contents of the haystack and the needle being searched for.

7. CONCLUSION

In conclusion, this project has successfully demonstrated the feasibility of implementing the Binary Search algorithm within the constraints of the 8085 microprocessor. We have achieved a deeper understanding of low-level programming and microprocessor architecture while showcasing the practicality of performing fundamental computational tasks on historic hardware.

Through our implementation, we have:

1. Translated the Binary Search algorithm into 8085 assembly language, efficiently utilizing the microprocessor's limited instruction set and memory capacity.

2. Overcome challenges related to memory management, data representation, and control flow, showcasing the adaptability of the 8085 microprocessor for algorithm execution.

3. Highlighted the algorithm's effectiveness by efficiently searching sorted datasets, affirming its time complexity of O(log n).

4. Recognized the historical significance of the 8085 microprocessor, which played a vital role in the early development of computing.

This project not only underscores the enduring value of fundamental algorithms but also serves as a testament to the ingenuity of early microprocessor developers. As we look forward, there are opportunities for further optimization and exploration of other algorithms, fostering a continued appreciation for the synergy of algorithms and hardware in the field of computer science.

8. REFERENCES

https://en.wikipedia.org/wiki/Binary_search_algorithm

https://www.sim8085.com

https://en.wikipedia.org/wiki/GNUSim8085

https://www.geeksforgeeks.org/8085-program-for-binary-search/