# Matrix Multiplication in 8085

MINI PROJECT REPORT

By

**Sanyog Dani (RA2211031010087)**

**Nikhil Kumar (RA2211031010097)**

**Arush Sirotiya (RA2211031010092)**

**V1 Section**

Under the guidance of   
**Dr. N. Senthamarai***In partial fulfilment for the Course*

Of

**21CSC201T – Computer Organization and Architecture**

in CSE with Specialization in IT

NWC Department



**FACULTY OF ENGINEERING AND TECHNOLOGY**

**SCHOOL OF NETWORK AN COMMUNICATION**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR (603203)**

**NOVEMBER 2023**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(Under Section 3 of UGC Act, 1956)**

**BONAFIDE CERTIFICATE**

**Certified that this B.Tech project report titled “Banker's Algorithm For Deadlock Detection And Avoidance”** is the bonafide work of

**Mr. Sanyog Dani [Reg. No.: RA2211031010087]**

**Mr. Arush Sirotiya [Reg. No.RA2211031010092]**

**Mr. Nikhil Kumar [Reg. No.: RA2211031010097]**

who carried out the project work under my supervision.

Certified further, that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

# SIGNATURE

**Dr. N. Senthamarai**

# Assistant Professor

**Department of Networking and Communications**

SRM Institute of Science and Technology

Kattankulathur

# 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 are highly thankful to my Course project Faculty **Dr. N. Senthamarai, Assistant Professor**, **Department of Networking and Communications,** for his/herassistance, timely suggestion, and guidance throughout the duration of this course project.

We extend my gratitude to our **HoD Dr. Annapurani Panaiyappan .K, Networking and Communication** 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.

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **S.No** | **CONTENTS** | **PAGE NO** |
| 1 | Objective | **5** |
| 2 | Abstract | **6** |
| 3 | Introduction | **7** |
| 4 | Hardware/Software Required | **7** |
| 5 | Concept/Working Principle | **9** |
| 6 | Program | **11** |
| 7 | Flowchart | **15** |
| 8 | Output | **17** |
| 9 | Result | **17** |
| 10 | Conclusion | **18** |
| 11 | References | **19** |

**OBJECTIVE**

The main objective of this mini project is to implement and demonstrate matrix multiplication on an 8085 microprocessor.

We aim to understand the hardware/software requirements, working principles, and methodology involved in this process.

**ABSTRACT**

This mini project explores the implementation of matrix multiplication using the 8085 microprocessor.

Matrix multiplication is a fundamental operation in linear algebra and has numerous applications in various fields.

By demonstrating how this operation can be performed on an 8085 microprocessor-based system, we gain valuable insights into computer organization and architecture.

**INTRODUCTION**

Multiplying two matrices of order m x n and n x p where m, n, p Z+ is an O(n3) where n is the maximum of m, n, p.

The project seeks to implement matrix multiplication for smaller order matrices on an Intel 8085 Microprocessor.

As you compile the program step by step using GNUSim 8085 Microprocessor you could visualize each row of the product matrix being filled.

As there is no direct multiplication operation available in 8085 Instructions, we intend to multiply numbers through repeated addition method using a loop.

In order to traverse through a row in Matrix 1 & a column in Matrix 2, we first load the starting address of row and column in stack and HL pair respectively.

For traversing through row and column we swap the values in HL register pair and top of stack and increment them.

We call multiplication sub-routine as and when we require multiplication of 2 numbers.

**HARDWARE/SOFTWARE REQUIREMENTS**

1. **Intel 8085 microprocessor kit or simulator:** The heart of the system is the Intel 8085 microprocessor, an 8-bit microprocessor that forms the central processing unit (CPU) for this project. It executes instructions and performs arithmetic and logical operations.
2. **Assembly language programming environment for 8085**
3. **Input Devices:** To provide input data, you'll need an interface to input matrices A and B into the system. This might involve using switches, a keypad, or another input method to enter the elements of the matrices into memory.
4. **Memory for data storage:** We need memory for data storage. The 8085 microprocessor uses memory for storing the program, data, and intermediate results. This includes Random Access Memory (RAM) to hold matrices A and B, the product matrix, and stack memory for subroutine calls**.**
5. **Output Devices:** To display the output, you require an output device. This can be LEDs, a monitor, or any other display method to show the resulting product matrix generated by the program.

**CONCEPT/WORKING PRINCIPLE**

**Algorithm for Matrix Multipication**

for ( int i = 0 ; i < rowNo ; i++ ){

for ( int j = 0 ; j < colNo ; j++ ){

for ( int k = 0 ; k < p ; k++ ){

result[i][j] = result[i][j] + first[c][k]\*second[k][d];

}

}

}

**Algorithm for Multiplication**

int number1, number2;

while( number2 != 0 ){

number1 = number1 + number2; number2--;

}

**Matrix Multiplication Algorithm for 8085 for 1 × n & n × 1**

Load HL pair with Address of 1st row and 1st column of Matrix1 Load Stack with Address of 1st row and 1st column of Matrix2 MVI E, 00H

Method : Load value in HL memory location in A register Load value of stack in B register

Call multiply subroutine to multiply two numbers

ADD E

STA E

INX H XCHG INX H

JMP Method

Store the value of E in specified memory Location

**Matrix Multiplication Algorithm for 8085 for 2 × 2 & 2 × 2**

Load C with 2

Load D with 2

Method1: DCR C

Method: Multiply row 1 vector with column 1 vector using algo defined above DCR D

if D != 0:

if C != 0: Load HL pair with add. of Matrix1[1][1]

Call Method

if C == 0: Load HL pair with add. of Matrix2[2][1]

Call Method

if D == 0: Load HL pair with add. of Matrix1[2][1]

MVI D,002H

if C == 0: HLT

if C != 0 : Call Method1

# PROGRAM

# Code for multiplication

# ; code for multiplication of

# ; two numbers by repeated

# ; addition

# ; two numbers to be multiplied

# ; are stored in 0002H and ; 0003H,

# ; output is stored in 0004H

# MOV B,0002H

# MOV C,0003H

# MVI A,00H

# LOOP: ADD B

# DCR C

# JNZ LOOP

# STA 0004H

# Multiplying row vector with column

# vector LXI H, 8500H

# PUSH 8508H

# Method: MOV M, A

# XCHG

# MOV M,B

# CALL MUL

# STA 8516H

# INX H

# XCHG

# INX H

# JMP Method

# Matrix Multiplication

# MVI C, 002H

# MVI D, 002H

# Method2: DCR C

# Method3: CALL MRC

# DCR D

# JNZ Method4

# Method4: ORI C, 00H

# JNZ Method5

# Method5: LXI H, 8500H

# JMP Method3

# ORI C, 00H

# JZ Method6: LXI H, 8508H

# JMP Method3

# ORI D, 00H

# JZ Method7:

# Method7: INX H, 8508H

# MVI D, 002H

# ORI C, 00H

# JNZ Method3

# ORI C, 00H

# JZ Method8

# Method8: HLT

# Final Code

# MVI C, 00

# LXI H, 8500

# LOOP2: LXI D, 8600

# CALL MUL

# MOV B,A

# INX H

# INX D

# INX D

# CALL MUL

# ADD B

# CALL STORE

# DCX H

# DCX D

# CALL MUL

# MOV B,A

# INX H

# INX D

# INX D

# ADD B

# CALL STORE

# MOV A,C

# CPI 04

# JZ LOOP1

# INX H

# JMP LOOP2

# LOOP1: HLT

# MUL: LDAX D

# MOV D,A

# MOV H,M

# DCR H

# JZ LOOP3

# LOOP4: ADD D

# DCR H

# JNZ LOOP4

# LOOP3: MVI H,85

# MVI D,86

# RET

# STORE: MVI B,87

# STAX B

# INR C

# RET

# FLOWCHART

*FIGURE 1: Multiplication of 2 numbers.*

# A diagram of a computer process Description automatically generated

*FIGURE 2: Multiplying row vector with column vector.*

# OUTPUT

**INPUT:**

(2050H) = 04H

(2051H) = 07H

(2052H) = 02H

(2053H) = 01H

**OUTPUT:**

(2054H) = 08H

(2055H) = 12H

(2056H) = 07H

(2057H) = O0H

# RESULT

# We have successfully computed Matrix multiplication of orders 1 *× n* & *n×* 1 and 2 *×* 2 & 2 *×* 2 and stored them in memory locations.

**CONCLUSION**

This mini project demonstrates the successful implementation of matrix multiplication on the 8085 microprocessor, showcasing the interplay between hardware and software.

Understanding how to perform complex mathematical operations on such a limited architecture provides valuable insights into computer organization and architecture.

This project contributes to a deeper comprehension of the 8085 microprocessor's capabilities and its role in handling real-world applications. It is also a stepping stone for students and enthusiasts to delve into the world of low-level programming and explore the synergy between hardware and software in computing systems.

**REFERENCES**

* [**https://www.wikipedia.org/**](https://www.wikipedia.org/)
* [**https://www.geeksforgeeks.org/**](https://www.geeksforgeeks.org/)
* [**https://www.w3schools.com/**](https://www.w3schools.com/)
* [**https://www.youtube.com/**](https://www.youtube.com/)