**IMPLEMENTATION OF MATRIX MULTIPLICATION USING ASSEMBLY LANGUAGE 8086**

**PROJECT REPORT**

**(2021 Regulation)**

***Submitted by***

**K.HARSHA VARDHAN (RA2211047010095)**

**ATHARV DOBHAL (RA2211047010134)**

**C.MANEESH KUMAR (RA2211047010094)**

**II Year/ III Semester**

**Academic Year: 2023 -2024**

***Under the Guidance of***

**Dr. A. Robert Singh**

**Assistant Professor, Department of Computational Intelligence**

***In partial satisfaction of the requirements for the degree of***

**BACHELOR OF TECHNOLOGY**

**in**

**ARTIFICIAL INTELLIGENCE**

**of**

**FACULTY OF ENGINEERING AND TECHNOLOGY**



S.R.M. Nagar, Kattankulathur, Chengalpattu District

NOVEMBER 2023

Logo, company name

Description automatically generated

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603203

**BONAFIDE CERTIFICATE**

Certified that this course project report titled **“IMPLEMENTATION OF MATRIX MULTIPLICATION USING ASSEMBLY LANGUAGE 8086”** is the bonafide work done by **K.HARSHA VARDHAN [RA2011047010095],C.MANEESH KUMAR [RA2211047010094], ATHARV DOBHAL[RA2211047010134]** of II Year/ III Sem B. Tech (AI) who carried out under my supervision for the course **Computer Organization and Architecture (21CSS201T)**. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

|  |  |
| --- | --- |
| Faculty In-Charge  **Dr. A Robert Singh**  Assistant Professor  Department of Computational Intelligence  SRM Institute of Science and Technology  Kattankulathur Campus, Chennai | Head of The Department  **Dr. R Annie Uthra**  Professor and Head  Department of Computational Intelligence  SRM Institute of Science and Technology  Kattankulathur Campus, Chennai |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No** | **CONTENTS** | **PAGE NO** |
| 1. | Abstract | **4** |
| 2. | Motivation/Challenges | **5-6** |
| 3. | Objective | **7** |
| 4. | Realistic Constraints and Deliverables | **8-9** |
| 5. | Algorithm | **10** |
| 6. | Program | **11-12** |
| 7. | Input&Output | **13-14** |
| 8. | Conclusion | **15** |
| 9. | References | **16** |

# Abstract:

This project presents the implementation of a 3x3 matrix multiplication algorithm using assembly language tailored for the Intel 8086 microprocessor. The Intel 8086, a widely recognized microprocessor of the x86 family, is known for its extensive instruction set and memory organization. This project delves into the world of low-level programming by combining mathematical computation and microprocessor-specific instruction sets to achieve a fundamental yet vital operation - matrix multiplication.

The primary objective of this project is to demonstrate a practical application of assembly language programming, specifically tailored to the Intel 8086 microprocessor. Assembly language, being a low-level programming language, allows for direct manipulation of the processor's registers and memory, offering a unique insight into the intricacies of computing at the hardware level.

Matrix multiplication is a fundamental operation widely used in scientific computing, computer graphics, and numerous other domains. It involves the multiplication of two matrices to produce a resulting matrix, necessitating complex arithmetic operations and memory management. The project not only demonstrates the proficiency of assembly language programming but also emphasizes the efficiency in handling complex mathematical computations on a specific hardware platform.

Efficient memory management and error-free execution are critical components of this project. The algorithm developed ensures proper handling of matrix data and intermediate results, reducing resource utilization to the minimum required. Error-checking mechanisms have been incorporated to guarantee the correctness of the multiplication and to handle boundary conditions effectively.

This project serves as both an educational resource and a practical demonstration of assembly language's versatility. It provides an opportunity to learn and understand the unique features of the Intel 8086 architecture, its instruction set, and memory organization. Furthermore, it offers insights into the broader application of assembly language, a skill that can be invaluable in understanding and optimizing low-level system operations.

In conclusion, this project showcases the power of assembly language for mathematical computations and emphasizes the significance of platform-specific programming. It serves as a learning resource, a reference for those interested in assembly language programming, and a practical demonstration of matrix multiplication on the Intel 8086 microprocessor. The implementation, along with well-documented code, offers an excellent starting point for exploring assembly language and its real-world applications in the realm of microprocessor programming.

# Motivation/Challenge:

In this report, we detail the implementation of matrix multiplication on the 8086 architecture, emphasizing the significance of this project in both educational and practical contexts. The following summary provides an overview of the key aspects explored in the subsequent sections:

**MOTIVATION:**

1. Computational Efficiency:

* Matrix multiplication is a fundamental operation in linear algebra and finds applications in various fields, such as computer graphics, scientific computing, and machine learning.
* Motivate the choice of implementing this operation on the 8086 by emphasizing the need for efficient computation on a specific hardware architecture.

2. Educational Purposes:

* Highlight the educational aspect of the project, demonstrating a practical application of assembly language programming.
* Emphasize how this project provides hands-on experience in understanding low-level hardware interactions and optimizing code for a specific processor.

3. Real-world Relevance:

* Discuss the real-world relevance of matrix multiplication in computing and how optimizing such operations at the assembly level can contribute to overall system performance.

**CHALLENGES:**

1. Limited Resources:

- Address the limitations of the 8086 architecture, such as its 16-bit architecture and limited register set. Explain how these constraints influenced the design and implementation of the matrix multiplication algorithm.

2. Memory Management:

- Discuss challenges related to memory management and data movement, considering the limited memory addressing capabilities of the 8086.

- Explain how you tackled issues related to efficient data storage and retrieval during the matrix multiplication process.

3. Optimization Trade-offs:

- Describe the trade-offs made during the optimization process. For example, balancing between code size and execution speed, and how certain optimizations were prioritized over others based on the architecture's strengths and weaknesses.

4. Instruction Set Limitations:

- Discuss any limitations or constraints imposed by the 8086 instruction set and how these influenced the choice of instructions for matrix multiplication.

- Highlight any creative solutions or workarounds implemented to overcome specific instruction set limitations.

5. Debugging and Testing:

- Address challenges related to debugging and testing assembly code on the 8086 architecture. Discuss the tools and methodologies used to identify and rectify errors.

6. Documentation and Understanding:

- Emphasize the importance of thorough documentation to enhance code readability and maintainability.

- Discuss how challenges related to understanding the intricacies of the 8086 architecture were addressed during the implementation.

**Objective:**

The primary objective of this project is to implement a 3x3 matrix multiplication algorithm using assembly language for the Intel 8086 microprocessor. This project aims to:

1. Demonstrate Proficiency: Showcase the ability to write efficient and functional assembly language code for a specific microprocessor architecture, the Intel 8086.

2. Mathematical Computation: Perform matrix multiplication as a fundamental mathematical operation, demonstrating how assembly language can handle arithmetic tasks.

3. Efficient Algorithm: Develop a well-optimized matrix multiplication algorithm that efficiently processes data with minimal resource utilization.

4. Memory Management: Learn and apply memory management techniques to store and manipulate matrix data and intermediate results during the multiplication process.

5. Error-Free Execution: Ensure that the implemented matrix multiplication algorithm produces accurate results and handles boundary conditions correctly.

6. Educational Resource: Create a project that serves as an educational resource, helping individuals understand the intricacies of assembly language programming and its practical application for mathematical operations.

7. Real-world Application: Illustrate the practicality and versatility of assembly language by using it to perform a matrix multiplication, a common operation in computer graphics, scientific computing, and many other fields.

8. Platform-specific Programming: Highlight the importance of writing code that is tailored to the Intel 8086 microprocessor architecture, taking advantage of its instruction set and memory organization.

9. Learning Opportunity: Provide a learning opportunity for individuals interested in assembly language programming and its role in low-level system operations.

10. Documented Implementation: Create a clear and well-documented project that can be used as a reference for those looking to understand and implement matrix multiplication in assembly language on the Intel 8086 microprocessor.

By accomplishing these objectives, this project aims to deliver a practical example of assembly language programming while showcasing the capability to perform matrix multiplication on the specific hardware platform of the Intel 8086.

Realistic Constraints and Deliverables:

## Realistic Constraints:

1. **Hardware Limitations:**

- The project operates within the constraints of the 8086 architecture, characterized by a 16-bit instruction set and a limited number of registers.

- The constraints of the hardware influenced decisions regarding algorithm design, memory usage, and optimization strategies.

2. **Memory Constraints:**

- Due to the limited memory addressing capabilities of the 8086, efficient memory management is paramount.

- The implementation addresses challenges related to storing and accessing matrices within the confines of the available memory.

3. **Instruction Set Limitations:**

- The project works within the boundaries of the 8086 instruction set, with considerations for its unique limitations and capabilities.

- Certain instructions may be more efficient or feasible than others for matrix multiplication, and these considerations impact the implementation.

## Deliverables:

**1. Optimized Matrix Multiplication Algorithm:**

- A robust and optimized algorithm designed specifically for matrix multiplication on the 8086 architecture.

- The algorithm balances computational efficiency with the constraints of the hardware.

**2. Assembly Language Code:**

- The deliverable includes well-documented assembly language code that implements the matrix multiplication algorithm on the 8086.

- The code adheres to best practices for readability and maintainability.

**3. Testing Suite and Results:**

- A comprehensive testing suite that validates the correctness and efficiency of the implemented matrix multiplication.

- Test results and performance metrics will be provided to demonstrate the effectiveness of the solution.

**4. Documentation:**

- Detailed documentation outlining the project's objectives, algorithm design, and implementation details.

- The documentation provides insights into the rationale behind specific design choices and optimizations.

# Algorithm:

# 1)Set pointers:

# -SI for the first input matrix.

# - DI for the second input matrix.

# - BP for the product matrix.

# 2)Initialize loop counters:

# - CL for elements in a row.

# - CH for elements in a column.

# - DH for temporary storage.

# 3)Outer loop (REPEAT3) for iterating through rows:

# - Copy column count to BL.

# 4)Middle loop (REPEAT2) for iterating through columns:

# - Initialize sum (DL) to zero.

# 5) Inner loop (REPEAT1) for performing multiplication and addition:

# - Get an element from the first row (AL).

# - Multiply it by the corresponding element from the second column.

# - Add the product to the sum.

# 6)Move pointers and decrement column count until DH is zero.

# 7)Store the sum in the product matrix.

# 8)Move pointers to the next element in the row.

# 9)Adjust pointers and decrement column count until BL is zero.

# 10)Move pointers to the first element of the second matrix and adjust loop counters.

# 11)Repeat the entire process until all rows are processed.

# 12) Stop the execution.

# This algorithm efficiently performs matrix multiplication for 3x3 matrices on the 8086 architecture, utilizing registers and pointers for data manipulation

# Program:

MOV SI,1301H ; SET SI AS THE POINTER FOR FIRST INPUT MATRIX

MOV DI,1401H ; SET DI AS THE POINTER FOR SECOND INPUT MATRIX

MOV BP,1501H ; SET BP AS THE POINTER FOR PRODUCT MATRIX

MOV CL,03H ; SET CL S COUNT FOR ELEMENTS IN A ROW

MOV CH,03H ; SET CH S COUNT FOR ELEMENTS IN A COLUMN

MOV DH,CH

REPEAT3:

MOV BL,DH ; COPY THE COLUMN COUNT IN BL REGISTER

REPEAT2:

MOV DL,00H ; INTIALIZE SUM AS ZERO

MOV CH,DH ; GET THE COULUMN COUNT IN DH

REPEAT1:

MOV AL,[SI] ; GET ONE ELEMENT OF THE ROW IN AL REGISTER

MUL [DI] ; GET THE PRODUCT OF ROW AND COLUMN ELEMENT IN AL

ADD DL,AL ; ADD THE PRODUCT TO SUM

INC SI ; INCREMENT THE FIRST INPUT MATRIX POINTER

ADD DI,03 ; LET DI POINTER TO NEXT ELEMENT OF SAME COLUMN OF 2ND MATRIX

DEC CH ; DECREMENT OF COLUMN COUNT

JNZ REPEAT1 ; REPEAT MULTIPLICATION AND ADDITION UNTIL DH COUNT IS ZERO

MOV [BP],DL ; STORE AN ELEMENT OF PRODUCT MATRIX IN MEMORY

INC BP ; INCREMENT THE PRODUCT MATRIX POINTER

SUB SI,03H ; MAKE SI TO POINT THE FIRST ELEMENT OF THE ROW

SUB DI,09H

INC DI

DEC BL ; DECREMENT COLUMN COUNT

JNZ REPEAT2

ADD SI,03H ; LET SI POINTER FIRST ELEMENT OF SECOND MATRIX

MOV DI,1401H ; MAKE DI TO POINT TO FIRST ELEMENT OF 2 ND MATRIX

DEC CL ; DECREMENT OF ROW COUNT

JNZ REPEAT3

HLT ; HALT AND EXECUTION

# PROGRAM IMAGE:

# C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\D0EC7DEB55FED8949EFB61CF1FA39004\WhatsApp Image 2023-10-31 at 19.18.18_80b92ac5.jpg

# Input:

**First matrix input**



**Second matrix input**



# Output:



# Conclusion:

# In conclusion, the implementation of matrix multiplication on the 8086 processor represents a successful venture into the intricacies of low-level programming and computational optimization. The chosen algorithm, programming language, and development tools were carefully considered to harness the full potential of the 8086 architecture. The performance evaluation revealed valuable insights into the execution time and memory usage, providing a quantitative measure of the algorithm's efficiency.

# Throughout the implementation, various challenges were encountered, ranging from the limitations of the 8086 architecture to debugging complexities. However, these obstacles served as learning opportunities, deepening our understanding of programming for specific hardware environments. Noteworthy optimizations were applied, significantly improving the overall speed and resource utilization of the matrix multiplication algorithm.

# This endeavor underscores the importance of tailoring algorithms to the nuances of a given architecture, showcasing the need for meticulous optimization in the realm of embedded systems. Looking ahead, the lessons learned from this implementation could pave the way for further advancements in optimizing algorithms for similar architectures or even inspire adaptations for diverse computational tasks.

# In essence, the successful implementation of matrix multiplication on the 8086 processor not only achieves the set project goals but also contributes valuable insights to the broader field of computer architecture, emphasizing the significance of efficient algorithms in maximizing the capabilities of specific hardware.

# References:

* Microprocessor Architecture, Programming, and Applications with the 8086 -

S Gaonkar

* 8080/8086 Assembly Language Programming Manual Copyright c 1977, 1978 Intel Corporation
* http://en.wikipedia.org/wiki/Matrix multiplication