



*Green University of Bangladesh*

*Department of Computer Science and Engineering (CSE)  
Semester: (Fall, Year: 2024), B.Sc. in CSE (Day)*

---

## **TaskMate:- Your Ultimate Task Assistant**

---

*Course Title: Microprocessor Lab  
Course Code: CSE-312  
Section: 221-D15*

### Students Details

<b>Name</b>	<b>ID</b>
Masum Hossain	221902164

*Submission Date: 18/11/2024  
Course Teacher's Name: Maisha Muntaha*

[For teachers use only: **Don't write anything inside this box**]

<u><b>Lab Project Status</b></u>	
<b>Marks:</b>	<b>Signature:</b>
<b>Comments:</b>	<b>Date:</b>

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Overview . . . . .	3
1.2	Motivation . . . . .	3
1.3	Problem Definition . . . . .	3
1.3.1	Problem Statement . . . . .	3
1.4	Objectives . . . . .	4
1.5	Application . . . . .	4
1.5.1	Educational Demonstrations . . . . .	4
1.5.2	Embedded Systems . . . . .	4
1.5.3	Legacy Systems . . . . .	4
1.5.4	Personal Productivity . . . . .	4
<b>2</b>	<b>Implementation of the Project</b>	<b>5</b>
2.1	Introduction . . . . .	5
2.2	Project Details . . . . .	5
2.2.1	Project Architecture . . . . .	5
2.3	Implementation . . . . .	6
2.3.1	Workflow . . . . .	6
2.3.2	Tools and Libraries . . . . .	6
2.4	Algorithms . . . . .	6
<b>3</b>	<b>Performance Evaluation</b>	<b>8</b>
3.1	Simulation Environment . . . . .	8
3.1.1	Hardware and Software Configuration . . . . .	8
3.2	Testing . . . . .	8
3.2.1	Add Task Functionality . . . . .	9
3.2.2	Update Task Functionality . . . . .	9

3.2.3	Delete Task Functionality . . . . .	10
3.2.4	Search Task Functionality . . . . .	10
3.3	Results Overall Discussion . . . . .	11
<b>4</b>	<b>Conclusion</b>	<b>12</b>
4.1	Discussion . . . . .	12
4.2	Limitations . . . . .	12
4.3	Scope of Future Work . . . . .	12

# Chapter 1

## Introduction

### 1.1 Overview

TaskMate is designed using assembly language. It provides a structured and interactive way to manage tasks with features such as adding, updating, searching, and deleting tasks. The software emphasizes simplicity and usability while showcasing the efficiency of low-level programming for real-time systems. By leveraging assembly language, the project also highlights how fundamental concepts of computer architecture can be applied to solve everyday problems efficiently.

### 1.2 Motivation

The motivation behind developing TaskMate stems from the increasing need for efficient and personalized task management tools. In a world driven by multitasking, many users, including students and professionals, rely on robust software to organize their work. While numerous applications are available, few focus on lightweight, resource-efficient systems. TaskMate was conceptualized to demonstrate how even low-level programming can create powerful applications, enabling users to experience a unique blend of performance and minimal resource consumption.

Additionally, the project serves as an educational endeavor to deepen understanding of assembly language, its practical applications, and the principles of structured programming.

### 1.3 Problem Definition

#### 1.3.1 Problem Statement

Task management in modern systems often relies on high-level, resource-heavy applications. These solutions, while functional, can be inefficient for constrained environments such as embedded systems or older devices. The problem addressed by TaskMate is to

develop a lightweight, low-resource-consuming task management tool using assembly language that can function effectively without modern overheads.

## **1.4 Objectives**

As TaskMate deals with real life scenario so the primary goals of this application are:

- To create a task manager that operates with minimal computational resources.
- To provide an intuitive interface for managing tasks while maintaining ease of use.
- To include core features such as adding, updating, deleting, and searching tasks.
- To serve as a practical demonstration of assembly language programming.
- To ensure tasks are displayed in a structured, chronological order based on task dates.

## **1.5 Application**

TaskMate has practical applications in various areas, particularly in environments with resource constraints. Some key applications include:

### **1.5.1 Educational Demonstrations**

It can be used as a teaching tool to showcase the potential of assembly language in practical software development.

### **1.5.2 Embedded Systems**

TaskMate's lightweight nature makes it suitable for integration into embedded devices that require task scheduling.

### **1.5.3 Legacy Systems**

The project is ideal for older hardware systems where modern applications may not run efficiently.

### **1.5.4 Personal Productivity**

Users who prefer minimalistic task management tools can leverage TaskMate for daily planning and organization.

In conclusion, TaskMate bridges the gap between low-level programming and real-world software needs, offering an efficient, engaging, and practical solution to task management. Through this project, it is evident that assembly language, despite its complexity, can be a powerful tool in modern application development.

# Chapter 2

## Implementation of the Project

### 2.1 Introduction

The implementation of the project revolves around using the principles of assembly language to develop a task scheduling tool. This chapter provides a detailed insight into the technical aspects, workflow, and methodologies adopted in creating *TaskMate*.

### 2.2 Project Details

In this section, we elaborate on the specifics of the project. Subsections detail the design, features, and structure of *TaskMate*.

#### 2.2.1 Project Architecture



Figure 2.1: Showing the project architecture.

## 2.3 Implementation

This section details the implementation phases, tools, libraries, and core workflow used in developing *TaskMate*. Screenshots, programming codes, and relevant explanations are provided.

### 2.3.1 Workflow

The workflow for implementing *TaskMate* is as follows:

- Step 1: Define and design the task management structure(ASCII2Design).
- Step 2: Implement task addition, update, and deletion functionality in assembly language.
- Step 3: Test the system on various hardware configurations to ensure compatibility and efficiency(EMU8086).

### 2.3.2 Tools and Libraries

The project primarily utilized the following:

- **Assembler:** EMU 8086 for writing and debugging assembly code.
- **Debugger:** Debugging tools for step-by-step code analysis.
- **Development Environment:** Integrated development environment for assembly programming.

## 2.4 Algorithms

Algorithms play a critical role in defining the efficiency and functionality of *TaskMate*. Below is the pseudo-code for task scheduling:

---

**Algorithm 1:** TaskMate Task Management Algorithm

---

**Data:** Task List  $T$ , User Input  $U$  (operation type, task details)

**Result:** Updated Task List  $T$  with Add, Update, or Delete operations

```
1 begin
2   while User does not choose EXIT do
3     Display menu options: {Add Task, Update Task, Delete Task, View
      Tasks, Exit};
4     switch User Input U do
5       case Add Task do
6         Task Description;
7         Add the task to Task List  $T$ ;
8         Display confirmation message;
9       case Update Task do
10        Prompt for Task ID to update;
11        if Task ID exists in T then
12          Prompt for new Task Name and/or Task Description;
13          Update the corresponding task in  $T$ ;
14          Display confirmation message;
15        else
16          Display error message: Task not found;
17      case Delete Task do
18        Prompt for Task ID to delete;
19        if Task ID exists in T then
20          Remove the task from  $T$ ;
21          Display confirmation message;
22        else
23          Display error message: Task not found;
24      case View Tasks do
25        Display all tasks in  $T$ ;
26      case Exit do
27        Terminate the program;
```

---





```

msg_add_header DB "*****", 0DH, 0AH
DB "*"
DB "*" ADD NEW task "*"
DB "*"
DB "*****", 0DH, 0AH
DB "", 0DH, 0AH
DB "$"

msg_delete_header DB "*****", 0DH, 0AH
DB "*"
DB "*" DELETE A task "*"
DB "*"
DB "*****", 0DH, 0AH
DB "", 0DH, 0AH
DB "$"

msg_updating_header DB "*****", 0DH,
0AH
DB "*"
DB "*" Updating a task "*"
DB "*"
DB "*****", 0DH, 0AH
DB "", 0DH, 0AH
DB "$"

msg_search_header DB "*****", 0DH,
0AH
DB "*"
DB "*" SEARCH A TASK "*"
DB "*"
DB "*****", 0DH, 0AH
DB "", 0DH, 0AH
DB "$"

msg_text DB "Enter Task description : $"
msg_date DB "Enter Task date(DD/MM/YYYY): $"
msg_added DB "Task added successfully...$"

msg_delete DB " DELETE functionality goes here $"
msg_empty DB "No task found$"

msg_update DB "Enter the SL No of the task you want to UPDATE$"
msg_option DB "[1] Main Menu [0] Exit$"
msg_back DB "OR Press [0] to back$"

msg_no_match DB "No matching task found.$"
msg_match_found DB "Matching Task Found:$"
msg_search_prompt DB "Enter to search$", 0
newline DB 13, 10, "$"

```

```

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MAINMENU_LOOP:
        CLEAR_SCREEN
        SHOW_HEADER

        INPUT_LOOP:
            GETC

            CMP AL, '1'
            JE SM1

            CMP AL, '2'
            JE SM2

            CMP AL, '3'
            JE SM3

            CMP AL, '4'
            JE SM4

            CMP AL, '5'
            JE SM5

            CMP AL, '0'
            JE END_MAIN

            JMP INPUT_LOOP

    SM1:
        CALL SHOW_ALL_TASKS
        JMP MAINMENU_LOOP

    SM2:
        CALL ADD_NEW_TASK
        JMP MAINMENU_LOOP

    SM3:
        CALL UPDATE_TASK
        JMP MAINMENU_LOOP

```

SM4:

CALL DELETE\_TASK  
JMP MAINMENU\_LOOP

SM5:

CALL SEARCH\_TASK  
JMP MAINMENU\_LOOP

END\_MAIN:

EXIT

MAIN ENDP

SHOW\_ENTERED\_TASKS PROC

SHOW newline  
CMP taskAdded, 0  
JE EMPTY  
SHOW\_TASK "1", todo\_text1, todo\_date1

CMP taskAdded, 2  
JGE SECOND\_TASK\_OUTPUT  
JMP CONT1

EMPTY:  
SHOW msg\_empty  
JMP CONT1

SECOND\_TASK\_OUTPUT:  
SHOW newline  
SHOW newline  
SHOW\_TASK "2", todo\_text2, todo\_date2  
JMP CONT1

; Point From EMPTY block  
CONT1:  
RET

SHOW\_ENTERED\_TASKS ENDP

ADD\_NEW\_TASK PROC

CLEAR\_SCREEN  
SHOW msg\_add\_header  
SHOW newline

SHOW msg\_text

```

MOV BL, taskAdded

CMP BL, 1
JE SECOND_TASK

    GET_STRING todo_text1
    SHOW newline

    SHOW msg_date
    GET_STRING todo_date1

    JMP CONT

SECOND_TASK:
    GET_STRING todo_text2
    SHOW newline

    SHOW msg_date
    GET_STRING todo_date2

    CALL SORT

CONT:
INC taskAdded
SHOW newline
SHOW newline
SHOW msg_added
SHOW newline
SHOW newline
SHOW msg_option
SHOW newline

GETC
RET

ADD_NEW_TASK ENDP

    DB "***** "0DH, 0AH
    DB "*"          All remaining procedures such as      "*", 0DH, 0AH
    DB "*"          Update Task,Delete Task,Search Task   "*", 0DH, 0AH
    DB "*****", 0DH, 0AH
    DB "", 0DH, 0AH
    DB "$"

    RET
SWAP_TODO ENDP
END MAIN

```

# Chapter 3

## Performance Evaluation

### 3.1 Simulation Environment

This section discusses the experimental setup and environment necessary for simulating the outcomes of the TaskMate project. It includes the hardware and software configurations, development tools, and methodologies utilized during the implementation and testing phases.

#### 3.1.1 Hardware and Software Configuration

The simulation was conducted on a system with the following specifications:

- **Processor:** Intel Core i5-10400
- **RAM:** 8 GB DDR4
- **Operating System:** Windows 10
- **IDE:** EMU 8086
- **Debugger:** Single steps and running the full program at once

The lightweight nature of **TaskMate** ensured compatibility with systems of varying capacities, including legacy hardware.

### 3.2 Testing

This section provides an analysis of the results obtained during the testing phase of **TaskMate**. Detailed evaluations of specific functionalities are highlighted, along with screenshots and graphical representations.

### 3.2.1 Add Task Functionality

The Add Task feature was tested with various inputs to validate its accuracy.

- **Scenario 1:** Adding details resulted in successful addition.
- **Scenario 2:** Inputting the wrong parameter produced an error message, ensuring data integrity.

```
*****
*                                     *
*               All of your TASKS    *
*                                     *
*****

1.Presentation
  10/12/2024

2.CT
  12/12/2024

[1] Main Menu      [0] Exit
_
```

Figure 3.1: Successful Addition of a Task

### 3.2.2 Update Task Functionality

The Update Task feature allowed modifications to existing tasks.

- **Scenario 1:** Updating a valid task ID successfully modified the task name and description.
- **Scenario 2:** Attempting to update a non-existent Task ID generated an appropriate error message.

```
*****
*                                     *
*               UPDATE task          *
*                                     *
*****

Enter the SL No of the task you want to UPDATE

1.ISD Presentation
  11/12/2024

2.CT
  12/12/2024

OR Press [0] to back
```

Figure 3.2: Successful updation of a Task .

### 3.2.3 Delete Task Functionality

The Delete Task feature effectively removed tasks from the list.

- **Scenario 1:** Deleting a valid Task ID successfully removed the entry from the task list.
- **Scenario 2:** Providing an invalid Task ID displayed a warning message.

```
*****
*                                     *
*                               SEARCH A TASK                               *
*                                     *
*****

Enter to search
CT
Matching Task Found:
  2.CT
    12/12/2024
OR Press [0] to back
```

Figure 3.3: Successful Searching of a Task

### 3.2.4 Search Task Functionality

The Search Task feature allows users to locate specific tasks within the TaskMate system.

- **Scenario 1:** Searching for an existing Task description fully successfully displays the corresponding task details.
- **Scenario 2:** Attempting to search for a non-existent Task details results in an error message, ensuring accurate search results.

```
*****
*                                     *
*                               DELETE A task                               *
*                                     *
*****

1.ISD Presentation
  11/12/2024
OR Press [0] to back
```

Figure 3.4: Successful Deletion of a Task.



### **3.3 Results Overall Discussion**

Overall, TaskMate successfully met its objectives, providing efficient task management in a resource-constrained environment. The testing phase revealed robust handling of edge cases, user errors, and invalid data inputs. The user-friendly interface and lightweight design make it suitable for various practical applications.

# Chapter 4

## Conclusion

### 4.1 Discussion

In this chapter, we summarize the work undertaken and the results achieved with the TaskMate project. The implementation of TaskMate showcases its practical applications in various environments, particularly in educational settings, embedded systems, and legacy systems. The project successfully demonstrates the potential of assembly language in modern software development, offering a minimalist yet efficient tool for task management. Through careful design and rigorous testing, TaskMate addresses the challenges of task scheduling in resource-constrained environments, providing a reliable solution for personal productivity and organizational needs.

### 4.2 Limitations

- **Limited Feature Set:** The tool lacks advanced features found in commercial task management applications, such as task prioritization, collaboration tools, and extensive reporting capabilities.
- **Interoperability Constraints:** TaskMate currently does not integrate seamlessly with other applications, limiting its usefulness in diverse software environments.

### 4.3 Scope of Future Work

- **Enhancing Interoperability:** Improving the ability of TaskMate to work with other software tools to facilitate better workflow integration.
- **User Interface Improvements:** Making the graphical user interface more intuitive and user-friendly to enhance the overall user experience.
- **Advanced Features:** Adding features like task prioritization, collaboration tools, and synchronization capabilities across devices.

- **Performance Optimization:** Further optimizing the code to reduce memory usage and enhance performance on older hardware by beep sound as well as adding priority based task Scheduling.

# References

*TaskMate: A Task Scheduling Application in Assembly Language*. Available at: <https://github.com/masum6268/microprocessorLab/tree/main/Project>

EMU8086 User Guide. *A Comprehensive Guide to Assembly Language Programming with EMU8086*. Available at: <https://www.emu8086.com/doc/index.html>

Paul Carter. *PC Assembly Language*. Available at: <https://pacman128.github.io/pcasm/>