



FINAL PROJECT REPORT

COURSE TITLE : Microprocessors and Microcontrollers Lab

COURSE CODE : CSE 360

Submitted To :

Md. Ismail
Lecturer,
Department of Computer Science and Engineering,
University of Information Technology & Sciences
(UITS)

Submitted By :

NAME : Tonny Talukder
ID Number : 0432220005101050
Section : 6A1
Batch : 52
Department : CSE

DATE OF SUBMISSION : 26/05/2025

Disk-Based Calculator for Two-Digit Arithmetic Operations using EMU8086

1. Introduction

This project is a **disk-based calculator** developed using **EMU8086 assembly language**. It performs basic arithmetic operations (addition, subtraction, multiplication, and division) on two-digit numbers and saves the result to a disk file.

2. Tools Used : EMU8086 Microprocessor Emulator

3. Objectives

- Design a simple text-based interface for user input.
- Implement basic arithmetic operations using assembly language.

4. Features

- User-friendly text interface
- Two-digit number operations
- Four operations: Addition, Subtraction, Multiplication, Division

5. Code Explanation

The code has the following main sections:

a. Input Section:

- First, the program displays the operation options (addition, subtraction, multiplication, division).
- Then, the program prompts the user to enter two two-digit numbers.

b. Operation Section:

- Based on the user's selection, the corresponding arithmetic operation is performed:
 - Addition: Sum of two numbers
 - Subtraction: Difference between two numbers
 - Multiplication: Product of two numbers
 - Division: Division of two numbers (with an error message for division by zero)

CODE :

```
.model small
.stack 100h

.data
    prompt1 db 'Select operation:', 0dh, 0ah
    prompt2 db '1. Addition', 0dh, 0ah
    prompt3 db '2. Subtraction', 0dh, 0ah
    prompt4 db '3. Multiplication', 0dh, 0ah
    prompt5 db '4. Division', 0dh, 0ah
    prompt6 db 'Enter your choice: $'
    prompt7 db 'Enter first number (2 digits): $'
    prompt8 db 'Enter second number (2 digits): $'
    result_msg db 0dh, 0ah, 'Result: $'
    error_msg db 0dh, 0ah, 'Error: Division by zero$'
    choice db 0
    num1 db 0
    num2 db 0
    result dw 0
    sign db '+'

.code
main:
    mov ax, @data
    mov ds, ax

    ; Display options
    mov ah, 09h
    lea dx, prompt1
```

int 21h

lea dx, prompt2

int 21h

lea dx, prompt3

int 21h

lea dx, prompt4

int 21h

lea dx, prompt5

int 21h

; Ask for choice

lea dx, prompt6

int 21h

mov ah, 01h

int 21h

sub al, '0'

mov [choice], al

; First number input (2 digits)

lea dx, prompt7

mov ah, 09h

int 21h

mov ah, 01h

int 21h

sub al, '0'

mov bl, al

mov ah, 01h

```
int 21h
sub al, '0'
mov bh, al
mov al, bl
mov ah, 0
mov bl, 10
mul bl
add al, bh
mov [num1], al
```

; Second number input (2 digits)

```
lea dx, prompt8
mov ah, 09h
int 21h
mov ah, 01h
int 21h
sub al, '0'
mov bl, al
mov ah, 01h
int 21h
sub al, '0'
mov bh, al
mov al, bl
mov ah, 0
mov bl, 10
mul bl
add al, bh
```

mov [num2], al

; Determine operation

mov al, [choice]

cmp al, 1

je addition

cmp al, 2

je subtraction

cmp al, 3

je multiplication

cmp al, 4

je division

addition:

mov al, [num1]

cbw

mov bl, [num2]

add al, bl

mov ah, 0

mov [result], ax

mov byte ptr [sign], '+'

jmp display_result

subtraction:

mov al, [num1]

mov bl, [num2]

```
sub al, bl
cmp al, 0
jge sub_positive
neg al
mov byte ptr [sign], '-'
jmp sub_store
```

sub_positive:

```
mov byte ptr [sign], '+'
```

sub_store:

```
mov ah, 0
mov [result], ax
jmp display_result
```

multiplication:

```
mov al, [num1]
mov bl, [num2]
mov ah, 0
mul bl    ; AX = AL * BL
mov [result], ax
mov byte ptr [sign], '+'
jmp display_result
```

division:

```
mov al, [num1]
```

```
mov ah, 0
mov bl, [num2]
cmp bl, 0
je division_error
div bl
mov ah, 0
mov [result], ax
mov byte ptr [sign], '+'
jmp display_result
```

division_error:

```
mov ah, 09h
lea dx, error_msg
int 21h
jmp exit_program
```

display_result:

```
mov ah, 09h
lea dx, result_msg
int 21h
```

; Show sign

```
mov dl, [sign]
mov ah, 02h
int 21h
```

; Show result


```
mov ax, [result]
call print_number
```

exit_program:

```
mov ah, 4Ch
int 21h
```

;----- Subroutine: print_number -----

print_number proc

```
; AX = number
; print two-digit or three-digit number
mov cx, 0
mov bx, 10
```

next_digit:

```
xor dx, dx
div bx      ; AX / 10 ? quotient in AX, remainder in DX
push dx     ; Store remainder (digit)
inc cx
cmp ax, 0
jne next_digit
```

print_loop:

```
pop dx
add dl, '0'
mov ah, 02h
int 21h
```

loop print_loop

ret

print_number endp

end main

OUTPUT :

The image shows a screenshot of an x86 assembly editor and emulator. The editor window displays assembly code for a calculator program. The code includes data definitions for prompts and messages, and a main routine that displays a menu, takes user input, and performs arithmetic operations. The emulator window shows the registers and a screen displaying the program's output.

```
edit: C:\Users\test\Desktop\project.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help

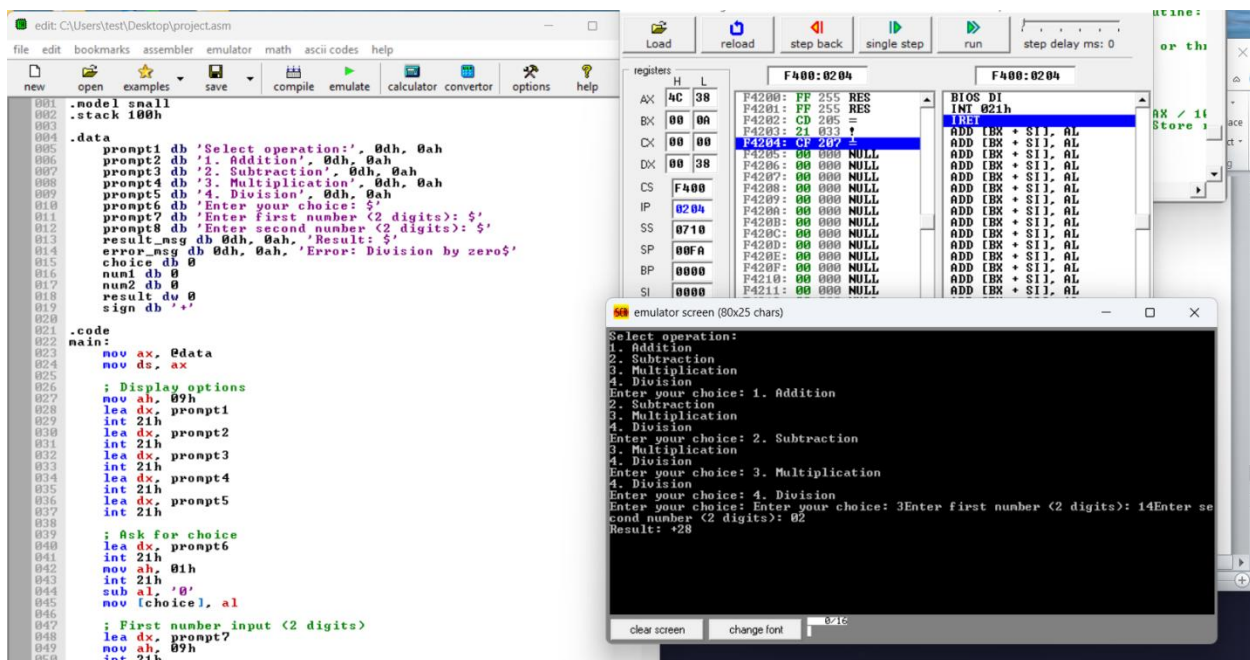
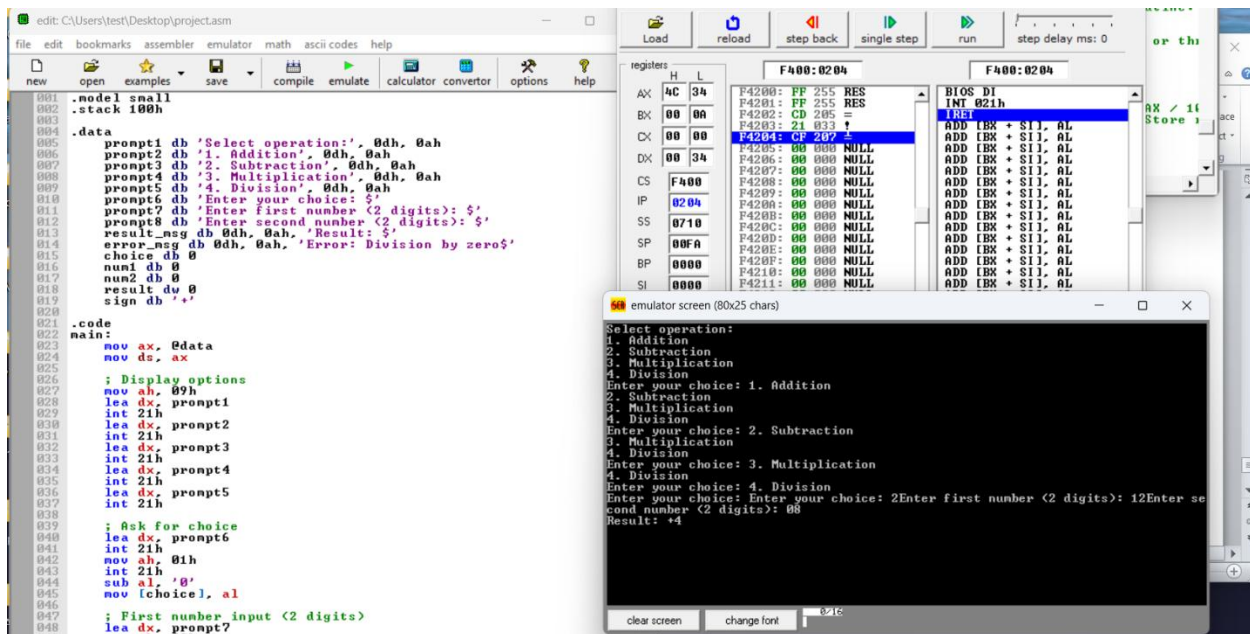
0001 .model small
0002 .stack 100h
0003
0004 .data
0005     prompt1 db 'Select operation:', 0dh, 0ah
0006     prompt2 db '1. Addition', 0dh, 0ah
0007     prompt3 db '2. Subtraction', 0dh, 0ah
0008     prompt4 db '3. Multiplication', 0dh, 0ah
0009     prompt5 db '4. Division', 0dh, 0ah
0010     prompt6 db 'Enter your choice: $'
0011     prompt7 db 'Enter first number <2 digits>: $'
0012     prompt8 db 'Enter second number <2 digits>: $'
0013     result_msg db 0dh, 0ah, 'Result: $'
0014     error_msg db 0dh, 0ah, 'Error: Division by zero$'
0015     choice db 0
0016     num1 db 0
0017     num2 db 0
0018     result dw 0
0019     sign db '+'
0020
0021 .code
0022 main:
0023     mov ax, @data
0024     mov ds, ax
0025
0026     ; Display options
0027     mov ah, 09h
0028     lea dx, prompt1
0029     int 21h
0030     lea dx, prompt2
0031     int 21h
0032     lea dx, prompt3
0033     int 21h
0034     lea dx, prompt4
0035     int 21h
0036     lea dx, prompt5
0037     int 21h
0038
0039     ; Ask for choice
0040     lea dx, prompt6
0041     int 21h
0042     mov ah, 01h
0043     int 21h
0044     sub al, '0'
0045     mov (choice), al
0046
0047     ; First number input <2 digits>
0048     lea dx, prompt7
0049     mov ah, 09h
```

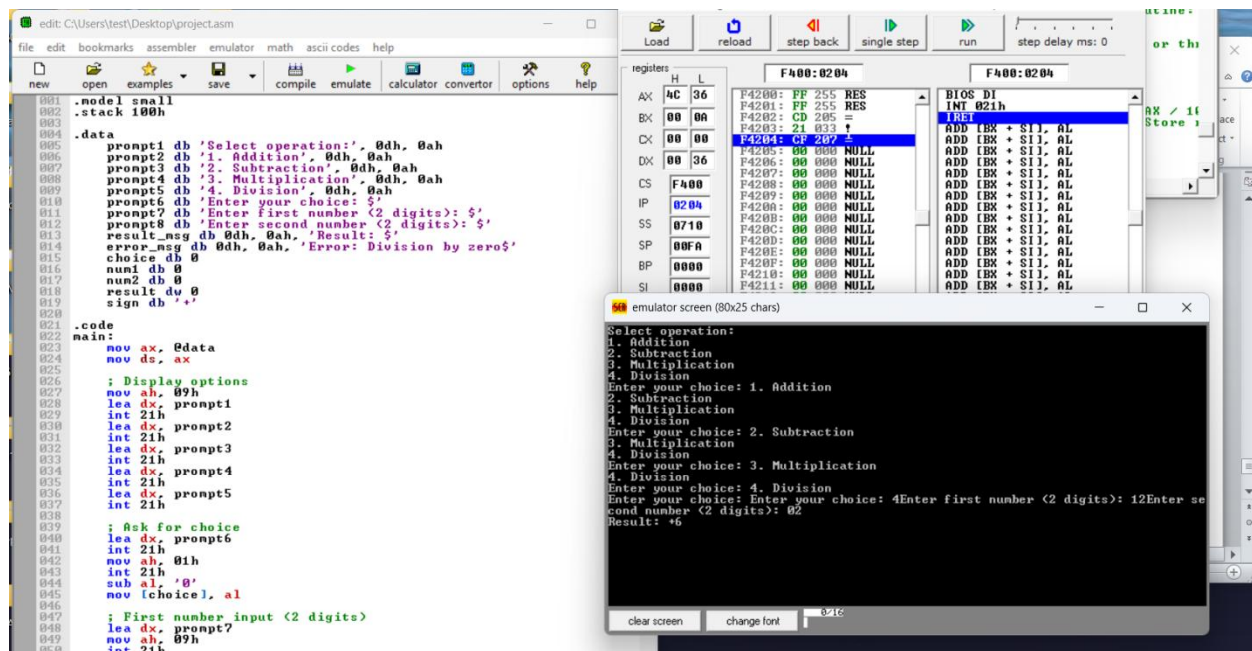
The emulator window shows the registers and a screen displaying the program's output. The registers window shows the following values:

Register	Value
AX	34
BX	00 00
CX	00 00
DX	00 34
SI	0000
SP	00F6
BP	0000
IP	0204
SS	0710

The screen displays the following output:

```
Select operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice: 1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice: 2. Subtraction
3. Multiplication
4. Division
Enter your choice: 3. Multiplication
4. Division
Enter your choice: 4. Division
Enter your choice: 1Enter first number <2 digits>: 12Enter se
cond number <2 digits>: 12
Result: *24
```





6. Challenges and Solutions

a. Challenges:

- Handling input and output in assembly language was challenging.
- Converting ASCII values to numbers and vice versa had to be done manually.

b. Solutions:

- A custom subroutine was written for ASCII-to-number conversion.
- Interrupts were used for input/output handling.

7. Conclusion

This project successfully demonstrates the creation of a disk-based calculator that allows users to perform basic arithmetic operations on two-digit numbers. It serves as a practical example of assembly language programming and disk file handling.