**Khulna University of Engineering & Technology**

****

**ECE 3104:Microprocessor & Microcomputer Laboratory**

**Subject: Design a simple calculator with 8086 microprocessor**

**Submitted To: Submitted By:**

Md. Minhajul Islam Arnab Sazin Ahmed , Roll:2009031

Lecturer Farjana Afroz, Roll:2009032

Department of ECE Md. Sumon Mia, , Roll:2009033

KUET, Khulna, Bangladesh Aditi Mondal, Roll:2009034

Md. Shafe-ul-Alam, Roll:2009035

Nawaz Talukder Sanglap Kaniz Fatema Mim, Roll:2009036

Lecturer

Department of ECE

KUET, Khulna, Bangladesh

**Objectives:**

* To design a simple calculator using 8086 microprocessor that can perform- addition, subtraction, multiplication and division with two 8-bit operands
* To simulate the calculator with necessary coding in emu8086.

**Introduction:**

8086 Microprocessor is an enhanced version of 8085 Microprocessor that was designed by Intel in 1976. It is a 16-bit Microprocessor having 20 address lines and16 data lines that provides up to 1MB storage. It consists of powerful instruction set, which provides operations like multiplication and division easily. Intel had designed the coprocessor 8087 that can do floating-point arithmetic and other complex mathematical operations. The 8086 can work in conjunction with 8087 to do both fixed-point, floating point and other complex mathematical functions. The 8086 works in a multiprocessor environment. Control signals for memory and I/O are generated by an external BUS controller. It can pre-fetch up to six instruction bytes from memory and queues them in order to speed up instruction execution. It requires 5V power supply and uses a 40-pin dual in line package. 8086 has two blocks- BIU and EU. The BIU performs all bus operations such as instruction fetching, reading and writing operands for memory and calculating addresses of the memory operands, prefetch of up to six bytes of instruction code. The instruction bytes are transferred to the instruction queue. The EU executes instructions from the instruction system byte queue.

An instruction is a binary pattern designed inside a microprocessor to perform a specific function. The entire group of instructions that a microprocessor supports is called Instruction Set. 8086 has more than 20,000 instructions. For instance-

* Data Transfer Instructions
* MOV Des, Src
* PUSH Operand
* POP Des
* LDS Des, Src
* Arithmetic Instructions
* ADD Des, Src
* SUB Des, Src
* MUL Src
* DIV Src
* INC Src
* Program Execution Transfer Instructions
* RET
* JMP Src

And so on.

**Theory:**

A calculator is designed with various arithmetic operations. Arithmetic operations in the 8086 microprocessor are facilitated through a repertoire of instructions tailored to handle addition, subtraction, multiplication, and division tasks. These instructions operate on data stored in registers or memory locations, employing diverse operand types to accommodate different data formats and computational requirements.

* Addition Instructions:
* **ADD D, S:** Adds the value of source operand S to the destination operand D.

Example: ADD AX, [2050] adds the value at memory address 2050 to the AX register.

* **ADC D, S:** Adds the value of source operand S and the previous carry flag to the destination operand D.

Example: ADC AX, BX performs addition with carry.

* Subtraction Instructions:
* **SUB D, S:** Subtracts the value of source operand S from the destination operand D.

Example: SUB AX, [SI] subtracts the value at the SI register from AX.

* **SBB D, S:** Subtracts the value of source operand S and the previous carry flag from the destination operand D.

Example: SBB [2050], 0050 performs subtraction with borrow.

* Multiplication Instructions:
* **MUL:** Multiplies the accumulator (AL) by an 8-bit register or memory operand.

Example: MUL BH multiplies AL by BH.

* **IMUL:** Performs signed multiplication with an 8-bit or 16-bit register.

Example: IMUL CX multiplies AX by CX (signed multiplication).

* Division Instructions:
* **DIV:** Divides the accumulator (AX) by an 8-bit register or memory operand.

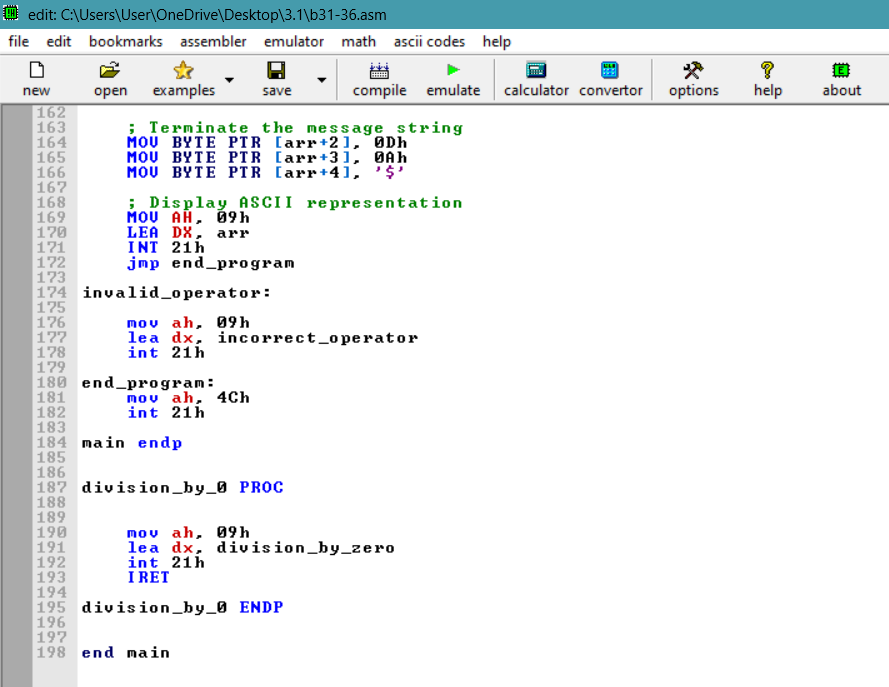
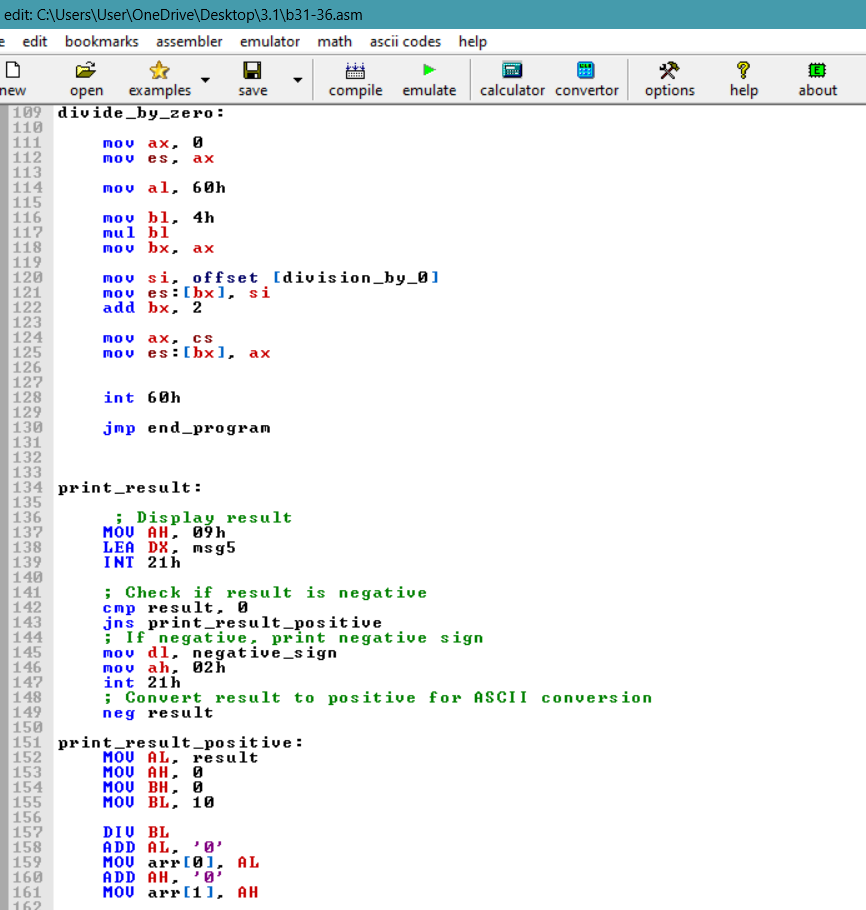
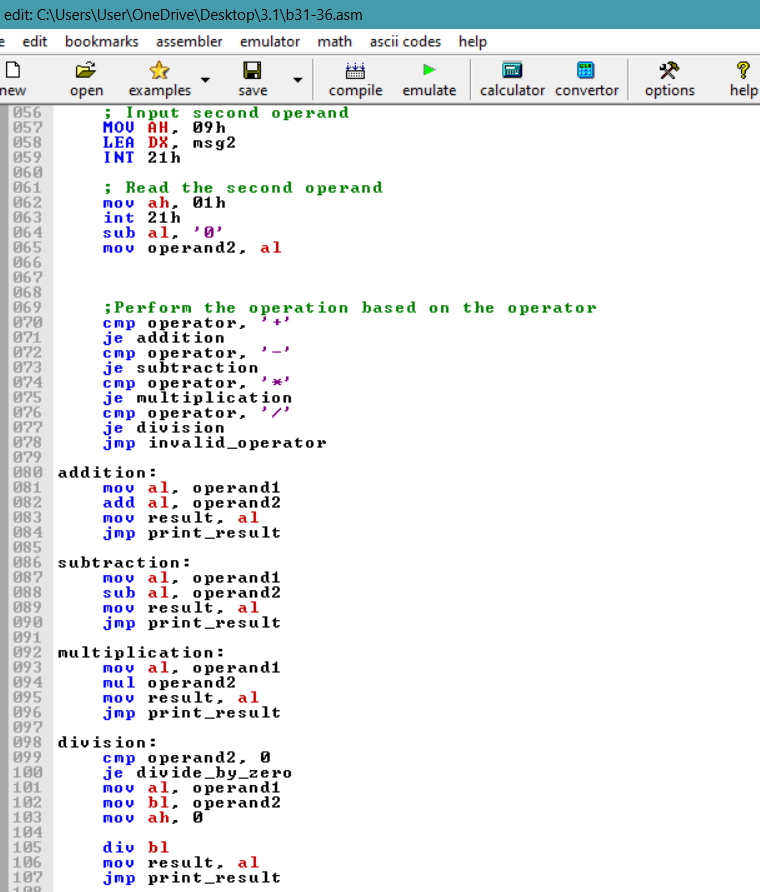
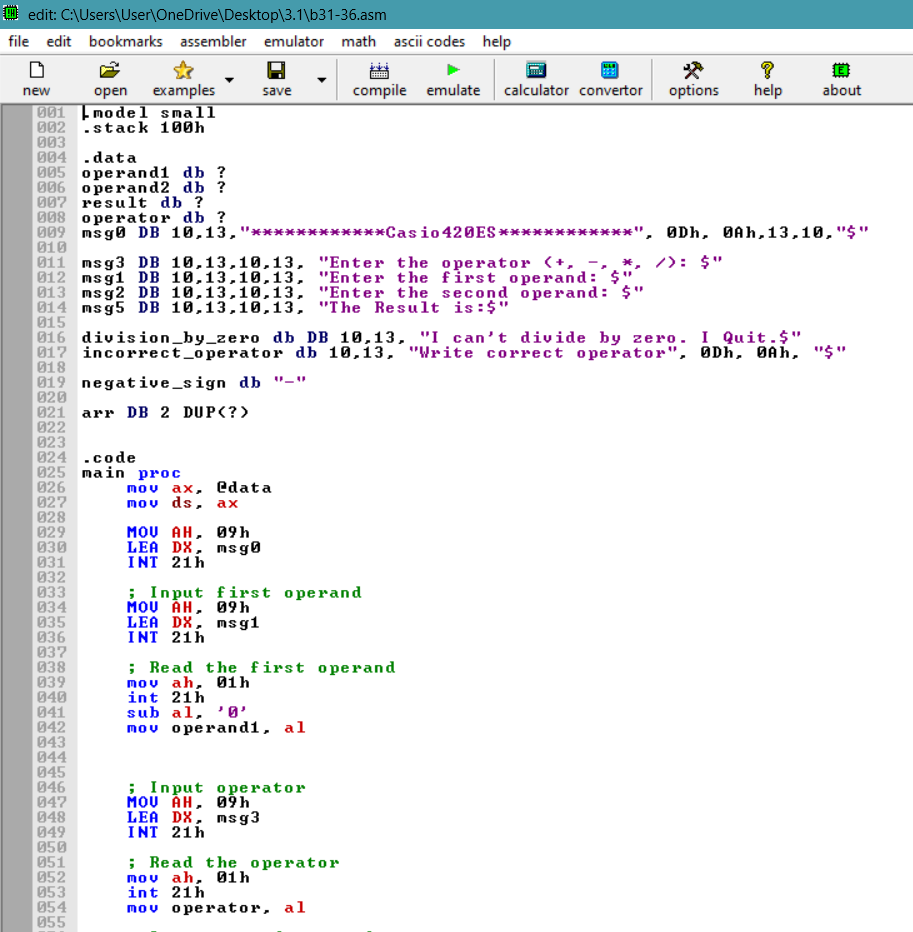
Example: DIV BL divides AX by BL, with quotient in AL and remainder in AH.

* **IDIV:** Performs signed division with an 8-bit or 16-bit register.

Example: IDIV BL divides AX by BL (signed division)

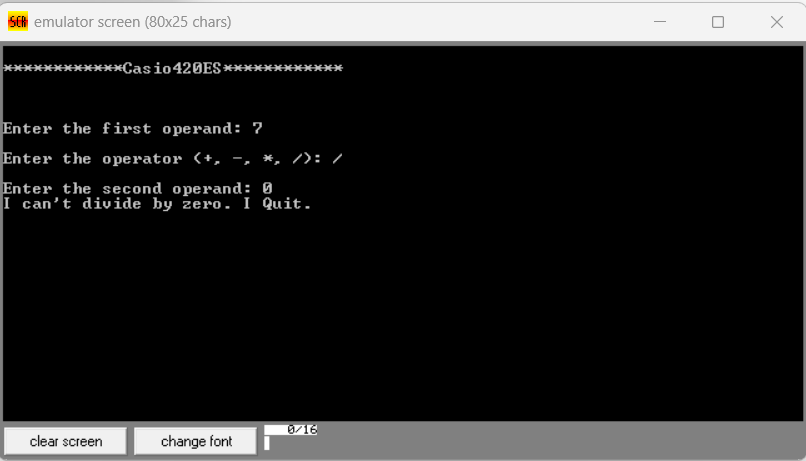
**Apparatus Required:**

1. Software- emu8086.

**Code:**

**Output:**



****

**Result Analysis:**

The project was simulated using emu8086, a software emulator for the 8086 microprocessor. Through emulation, the functionality of the calculator was thoroughly tested to ensure it performs the intended operations accurately and displays the correct results. Specifically, each arithmetic operation—addition, subtraction, multiplication, and division—was tested with various input values to validate their correctness.

The program uses various messages (MSG1 to MSG5) to guide users and stores input numbers (NUM1, NUM2) and the result of the operation (RESULT). To interact with users, the program relies on interrupt 21h with service number 9 for displaying messages and service number 1 for input from the keyboard. Initially, the program sets up the data segment and presents a series of messages to prompt users to select an operation. Users' choices are obtained from the keyboard (MOV AH, 1 / INT 21H), and the ASCII value of the chosen operation is converted to its numeric equivalent by subtracting 48 (SUB BH, 48). Based on the user's choice, the program jumps to one of the labels ADD, SUB, MUL, or DIV, each containing similar structures.

In each operation, users input the first number (MSG1) and second number (MSG2), which are then converted to their numeric equivalents. The program performs the chosen arithmetic operation (ADD, SUB, MUL, or DIV), stores the result in RESULT, converts the result back to ASCII characters, and displays it (MSG5). Afterward, the program jumps to the END\_PROGRAM label to clean up before exiting.

If users attempt to divide by zero, the program jumps to the DIVISION\_BY\_ZERO\_ISR section. This section handles division by zero errors by displaying an error message (MSG4) and then jumping to END\_PROGRAM. Finally, the END\_PROGRAM section prints a final message (MSG) and jumps to EXIT to terminate the program.

**Discussion:**

This project exemplifies the real-world utilization of the 8086 microprocessor in executing fundamental mathematical operations. Through user interaction and error management mechanisms, it highlights the foundation for constructing more sophisticated and user-friendly applications. Additionally, it prompts consideration of the constraints imposed by a 16-bit architecture when dealing with extensive calculations, emphasizing the necessity for emulation tools during development. During the development of this project, several challenges were encountered. Primarily, the code was unable to handle two digit numbers due to limitations in the method used for converting ASCII characters to numeric values. Additionally, there was no support for printing negative numbers, restricting the program's usability. Furthermore, the code did not effectively display the interrupt vector table, which could have aided in debugging and understanding system behavior. By emphasizing practical applications and acknowledging limitations, this project not only demonstrates the capabilities of the 8086 microprocessor but also encourages further exploration and innovation within the realm of assembly programming.

**Conclusion:**

This project marks a successful endeavor in developing a calculator through 8086 microprocessor coding. Throughout the process, we gained a comprehensive understanding of the various instructions and their applications, particularly in handling console input within the 8086 architecture. Additionally, we delved into the concepts of software and hardware interrupts, effectively utilizing interrupt service routines within our code. In summary, this project showcases our ability to put theoretical knowledge into practice, strengthening our skills in embedded system development and microprocessor programming.

**References:**

* https:// www.geeksforgeeks.org.com
* <https://www.academia.edu/8057299/An_8_Bit_Scientific_Calculator_Based_Intel_8086_Virtual_Machine_Emulator>