Name:	Sakshi Prabhakar Patil
Roll No:	44
Class/Sem:	SE/IV
Experiment No.:	1
Title:	To perform basic arithmetic operations on 16-bit data.
Date of Performance:	
Date of Submission:	
Marks:	
Sign of Faculty:	

Aim: Assembly Language Program to perform basic arithmetic operations (addition, subtraction, multiplication, and division) on 16-bit data.

Theory:

MOV: MOV Destination, Source.

The MOV instruction copies data from a specified destination. word or byte of data from a specified destination.

Source: Register, Memory Location, Immediate Number

Destination: Register, Memory Location

MOV CX, 037AH; Put immediate number 037AH to CX.

ADD: ADD Destination, Source.

These instructions add a number source to a number from some destination and put the result in the specified destination.

Source: Register, Memory Location, Immediate Number

Destination: Register, Memory Location

The source and the destination in an instruction cannot both be memory locations.

ADD AL, 74H; add the immediate number to 74H to the content of AL. Result in AL.

SUB: SUB Destination, Source.

These instructions subtract the number in some source from the number in some destination and put the result in the destination.

Source: Immediate Number, Register, or Memory Location.

Destination: Register or a Memory Location.

The source and the destination in an instruction cannot both be memory locations.

SUB AX, 3427H; Subtract immediate number 3427H from AX.

MUL: MUL Source.

This instruction multiplies an unsigned byte from some source times an unsigned byte in the AL register or an unsigned word from some source times an unsigned word in the AX register.

Source: Register, Memory Location.

MUL CX; Multiply AX with CX; result in high word in DX, low word in AX.

DIV: DIV Source.

This instruction is used to divide an unsigned word by a byte or to divide an unsigned double word (32 bits) by a word.

Source: Register, Memory Location.

If the divisor is 8-bit, then the dividend is in AX register. After division, the quotient is in AL and the remainder in AH.

If the divisor is 16-bit, then the dividend is in DX-AX register. After division, the quotient is in AX and the remainder in DX.

DIV CX; divide double word in DX and AX by word in CX; Quotient in AX; and remainder in DX.

Algorithm to add two 16-bit numbers

1. Load the first number in AX
2. Load the second number in BX

3 Add the second number to AX

4. Store the result in AX.

Algorithm to subtract two 16-bit numbers

1. Load the first number in AX.
2. Load the second number. in BX 3. Subtract the second number AX

4. Store the result in AX.

Algorithm to multiply a 16-bit number by an 8-bit number

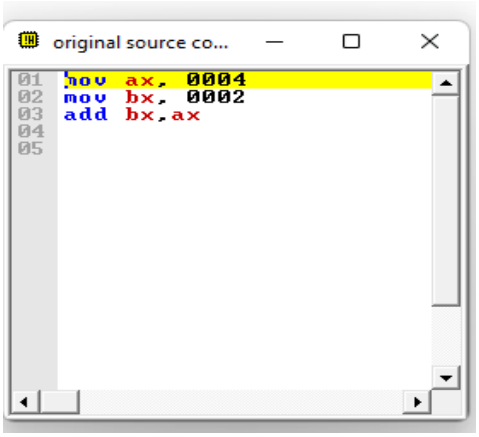
1. Load the first number in AX.
2. Load the second number. in BL
3. Multiply DX and AX.
4. The result is in DX and AX.

Algorithm to divide a 16-bit number by an 8-bit number

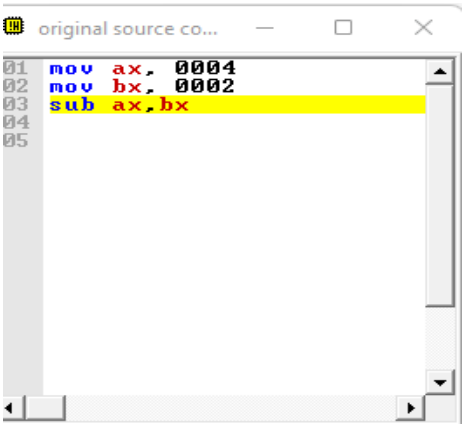
1. Load the first number in AX.
2. Load the second number. in BL
3. Divide AX by BL.
4. After division, the quotient is in AL and the remainder is in AH.

Assembly code:

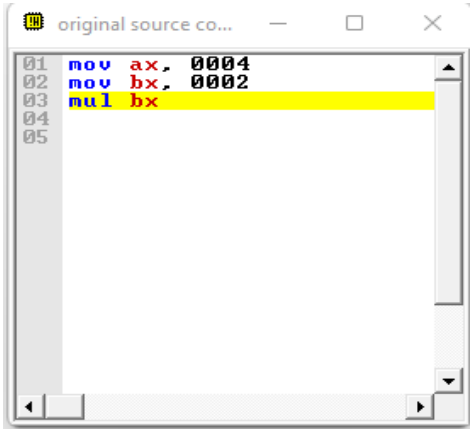
1. Add



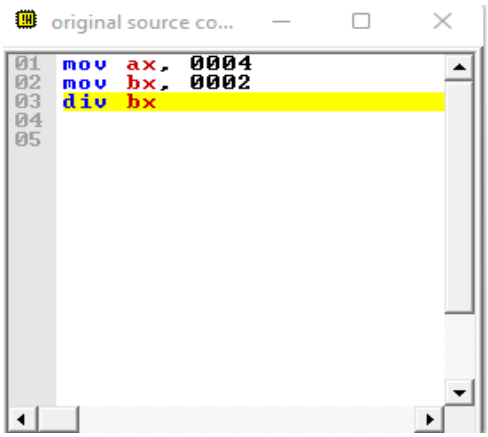
2. Subtract



3. Multiply

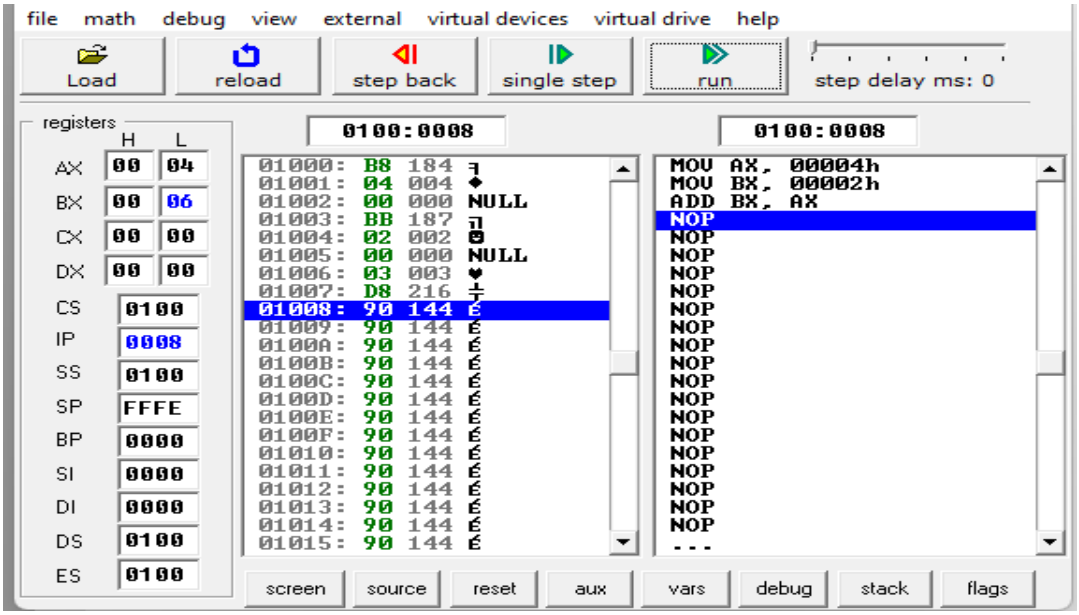


4. Division

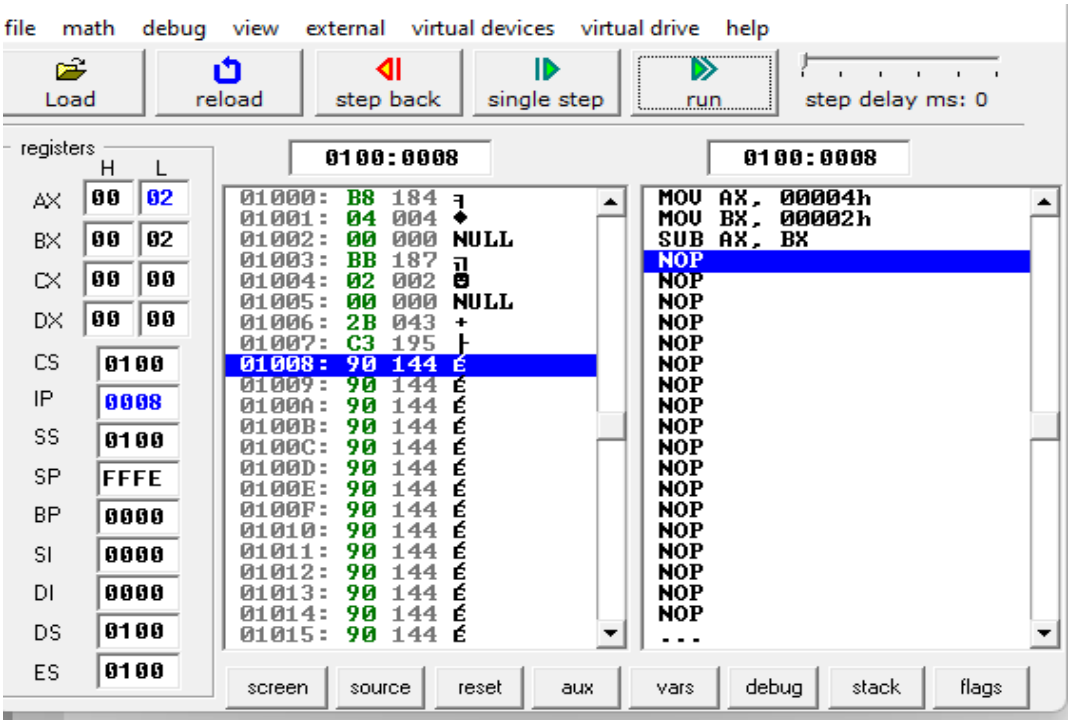


Output:

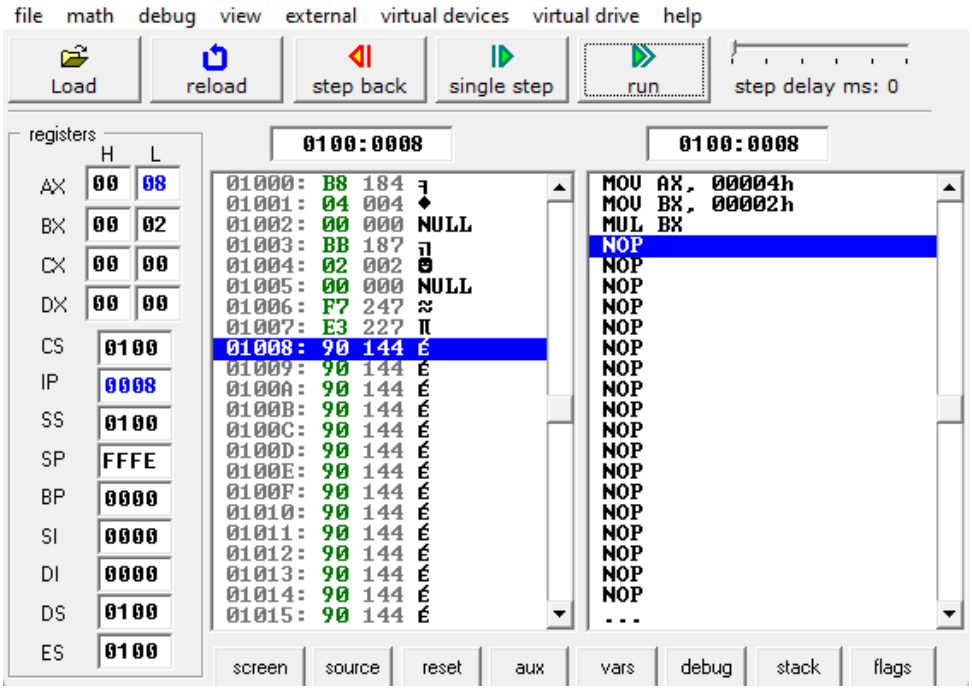
1. Add



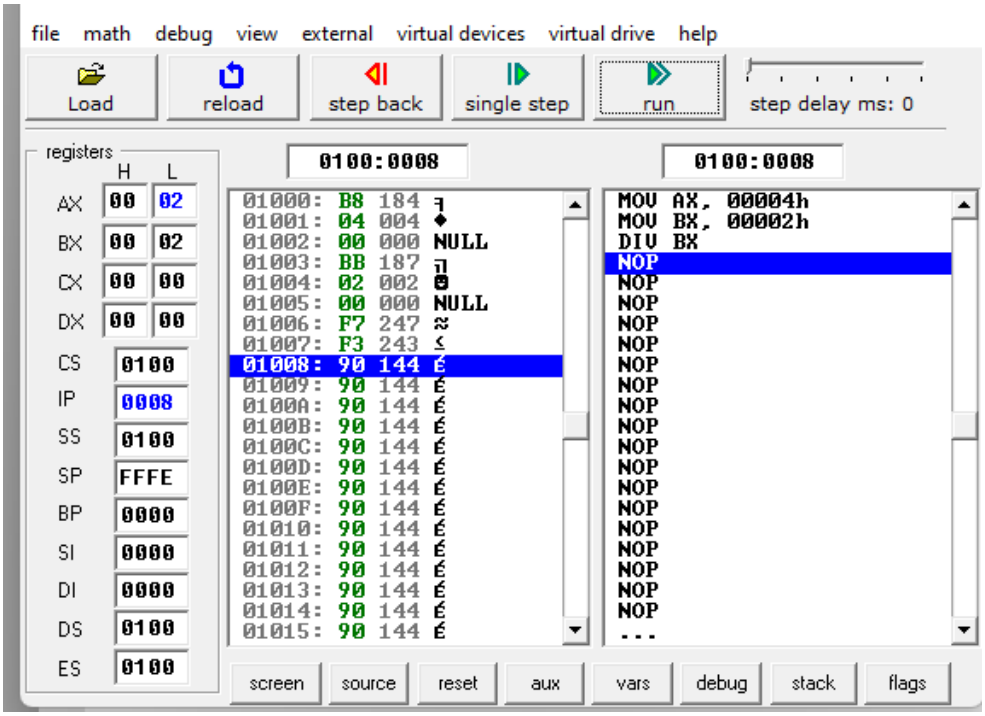
2. Subtract



3. Multiply



4. Division



Conclusion:

1. Explain the features of 8086.

Ans. The Intel 8086, a 16-bit microprocessor, revolutionized computing with its features. It boasts a 16-bit architecture, allowing it to process data in 16-bit chunks for enhanced performance. With a clock speed up to 5 MHz, it executes instructions swiftly. Its 20-bit address bus can access up to 1 MB of memory, facilitating large-scale applications. The 8086 supports a rich instruction set, including arithmetic, logical, and control operations, enabling versatile computing tasks. It incorporates powerful segmentation and addressing modes for efficient memory management. Additionally, its compatibility with a wide range of peripherals and software makes it a cornerstone in the evolution of personal computing.

2. Explain general purpose and special purpose registers.

Ans. General-purpose registers (GPRs) are registers within a computer processor that can be used for a wide variety of purposes, hence the name "general-purpose." These registers are typically used to store data temporarily during program execution. They are versatile and can be employed by the programmer for tasks such as arithmetic operations, data manipulation, memory addressing, and holding intermediate results.

Special-purpose registers, on the other hand, are registers that are designed for specific functions within the processor. These registers serve dedicated purposes and often have predefined roles in the execution of instructions or in managing the processor's operation. Examples of special-purpose registers include the program counter (PC), which holds the memory address of the next instruction to be executed; the instruction register (IR), which holds the currently fetched instruction; and the status register (flags register), which stores information about the outcome of arithmetic or logical operations, such as whether the result is zero or negative.

In summary, while general-purpose registers offer flexibility and can be used for various tasks, special-purpose registers serve specific functions critical to the operation of the processor and execution of instructions.