Name:	Sarvesh Surve
Roll No:	73
Class/Sem:	SE/IV
Experiment No.:	1
Title:	To perform basic arithmetic operations on 16-bit data.
Date of Performance:	10/01/2024
Date of Submission:	24/01/2024
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 NumberDestination:
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 NumberDestination:

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

Load the first number in AX

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

Load the first number in AX.

Load the second number. in BX 3. Subtract the second number to

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.

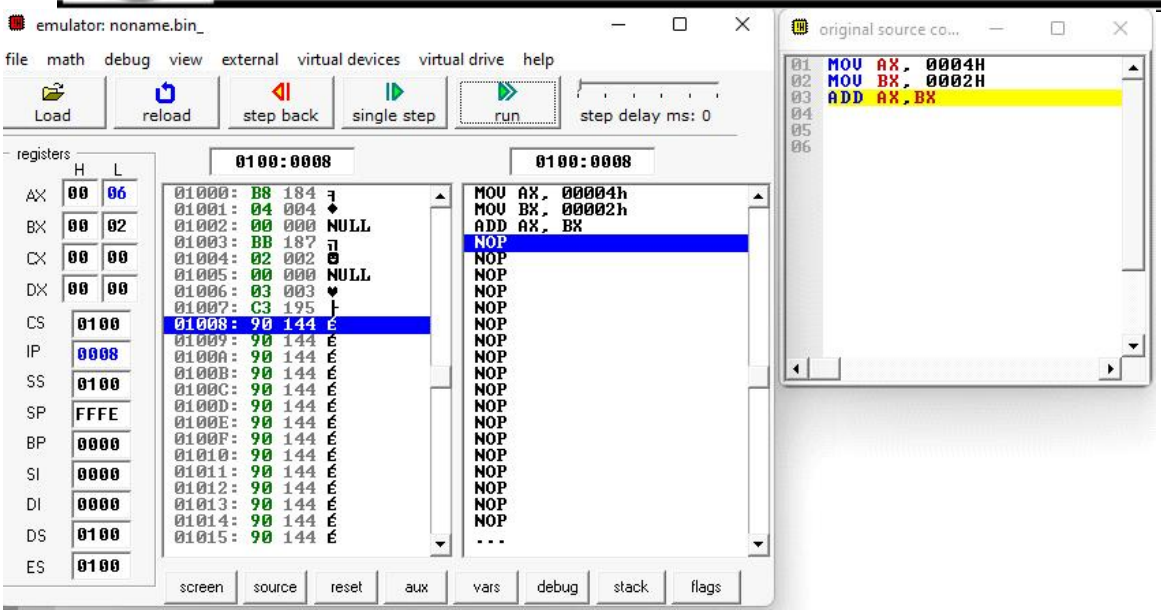
Code & Output:

Addition

```
MOV AX, 0004H

MOV BX, 0002H

ADD AX,BX
```

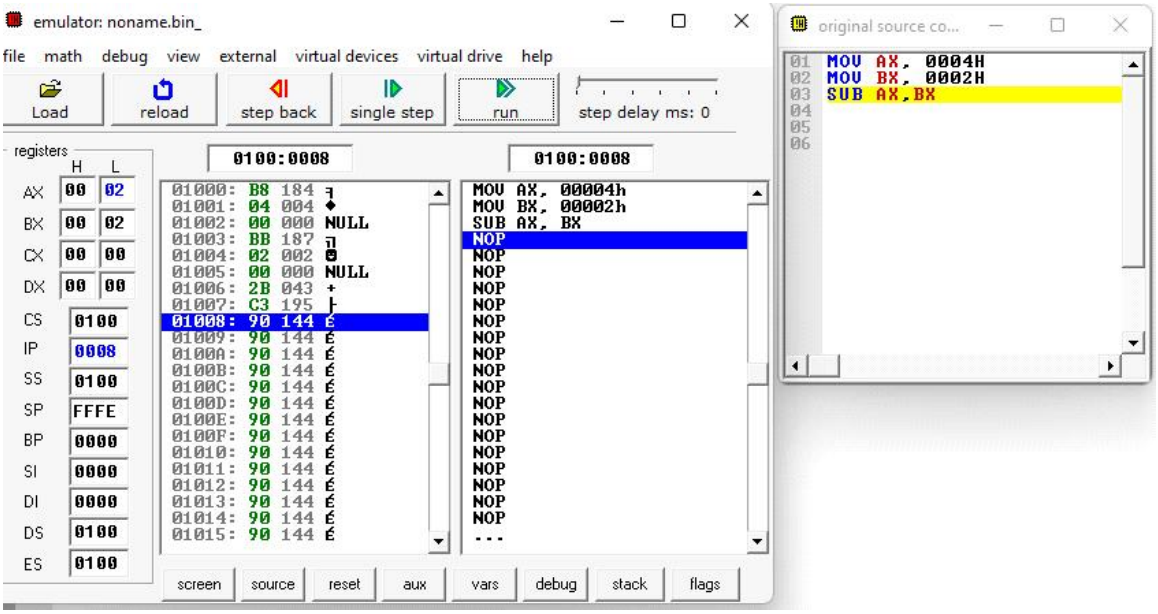


Subtraction

MOV AX, 0004H

MOV BX, 0002H

SUB AX,BX

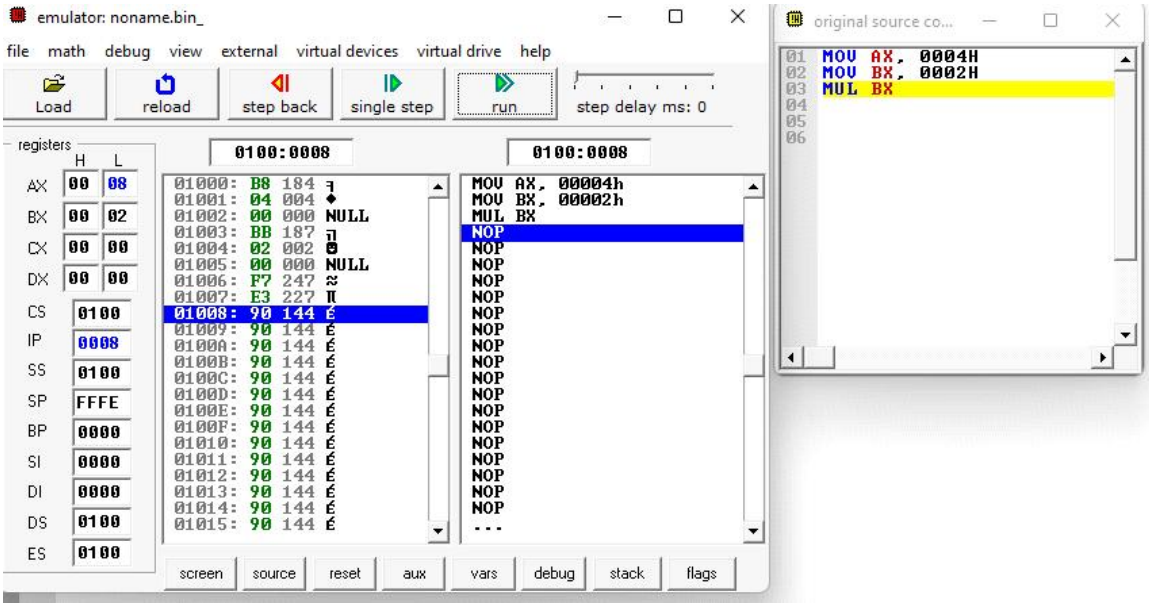


Multiplication

MOV AX, 0004H

MOV BX, 0002H

MUL BX

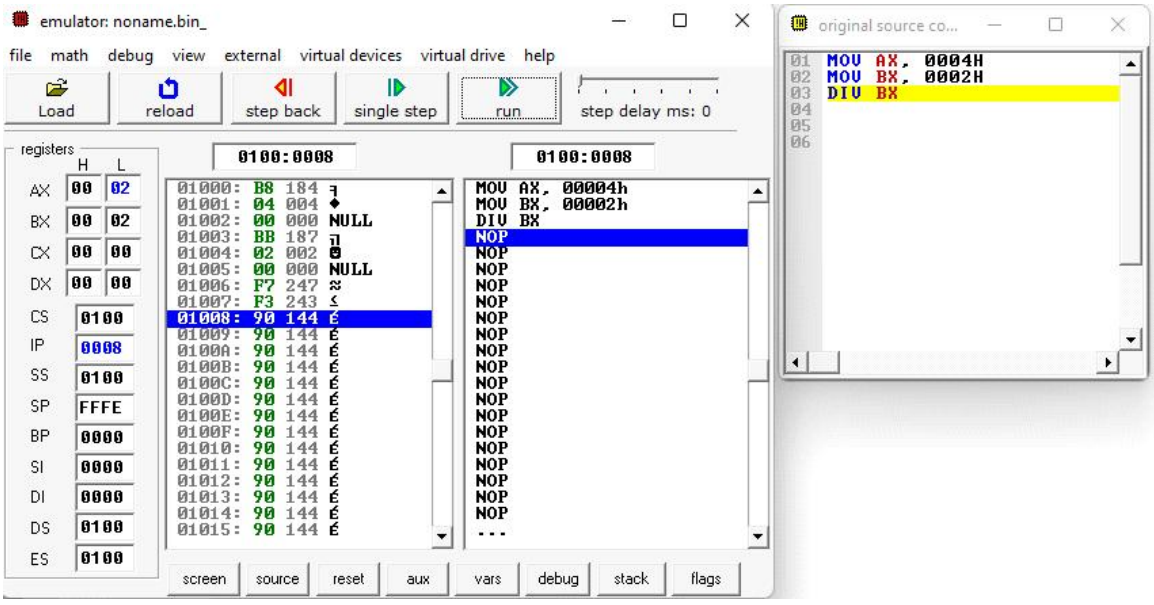


Divison

MOV AX, 0004H

MOV BX, 0002H

DIV BX



Conclusion:

Assembly language facilitates direct hardware-level arithmetic operations. Using MOV, ADD, SUB, MUL, and DIV instructions, 16-bit addition, subtraction, multiplication, and division are performed efficiently. These operations involve loading operands into registers, executing the operation, and storing results back, essential for low-level programming and optimization.

1. Explain the features of 8086.

The 8086 microprocessor, launched by Intel in 1978, features:

16-bit architecture: Processes data and addresses in 16-bit chunks, allowing for larger memory addressing and more extensive data manipulation.

Segmented memory: Utilizes segment registers to access memory beyond the 64KB limit, enabling addressing up to 1MB.

Instruction set: Supports a wide range of instructions for arithmetic, logical, and control operations, facilitating versatile programming.

Multipurpose registers: Includes general-purpose registers like AX, BX, CX, DX, along with segment registers and special-purpose registers for addressing and control.

Interrupt handling: Provides mechanisms for responding to external events, allowing the processor to pause current tasks and handle critical events.

Efficient execution: Executes instructions in multiple stages, including fetch, decode, execute, and memory access, optimizing performance.

Backward compatibility: Serves as the foundation for the x86 architecture, ensuring compatibility with later processors while enabling legacy software support.

2. Explain general purpose and special purpose registers.

In the 8086 microprocessor:

General-purpose registers:

- AX (Accumulator): Used for arithmetic and data manipulation operations. Often holds operands and results of arithmetic operations.
- BX (Base): Typically used in addressing modes for array indexing and data movement.
- CX (Count): Primarily used for iteration and loop control in repetitive operations.
- DX (Data): Often used in conjunction with AX for extended precision arithmetic and I/O operations.

Special-purpose registers:

- IP (Instruction Pointer): Points to the memory address of the next instruction to be executed.
- SP (Stack Pointer): Points to the top of the stack, used for managing subroutine calls and data storage.
- BP (Base Pointer): Typically used as a reference point for accessing parameters and local variables within procedures.
- SI (Source Index) and DI (Destination Index): Used in string operations for source and destination addressing, respectively.
- FLAGS: Stores status and control flags such as carry, zero, sign, and overflow, indicating the outcome of arithmetic and logical operations.