Name:	Pranav Shetty
Roll No:	53
Class/Sem:	SE/IV
Experiment No.:	1
Title:	To perform basic arithmetic operations on 16-bit data.
Date of Performance:	23/01/2024
Date of Submission:	30/01/2024
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 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 to AX

4. Store the result in AX.

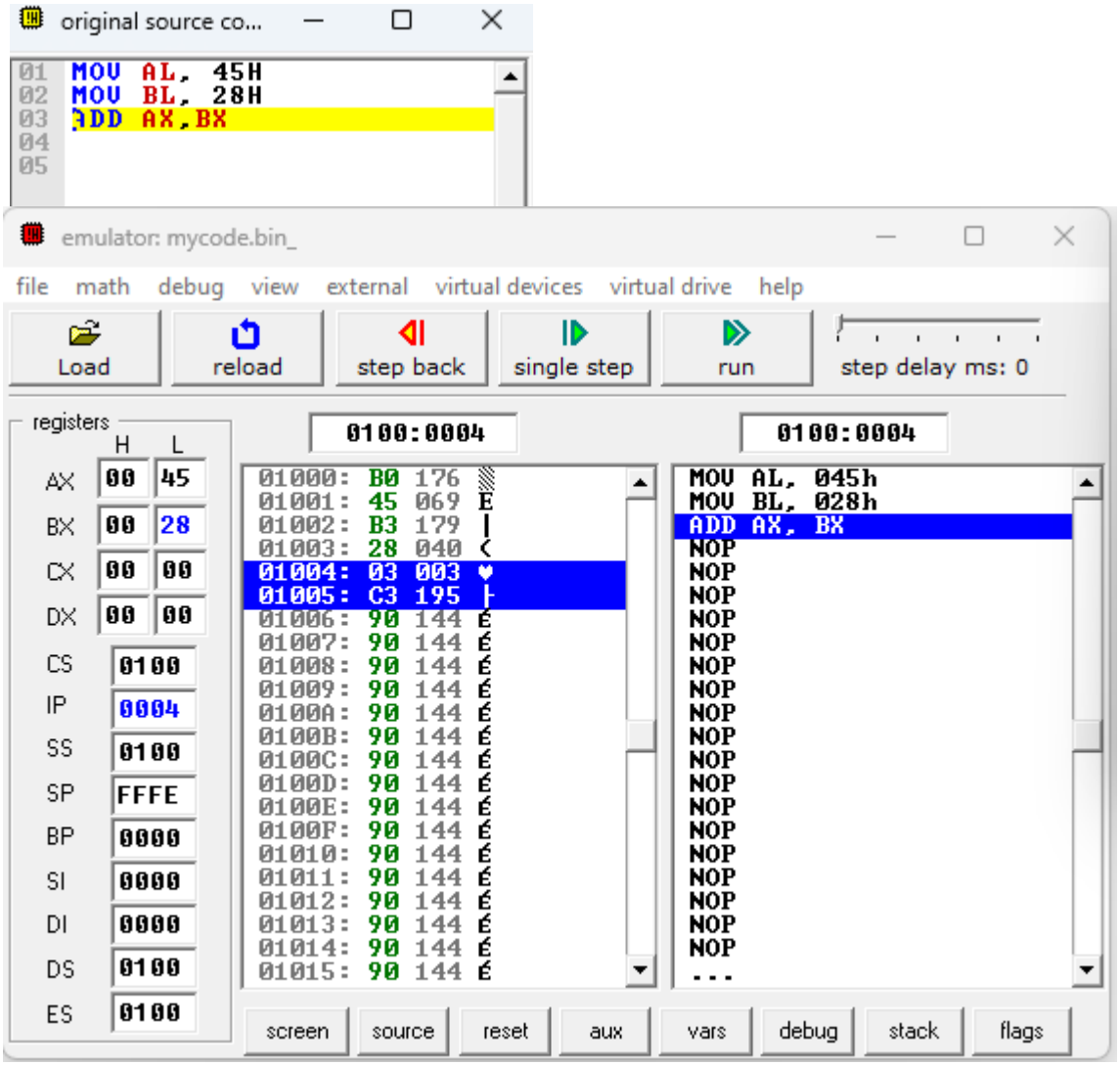
Algorithm to multiply a 16-bit number by an 8-bit number

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

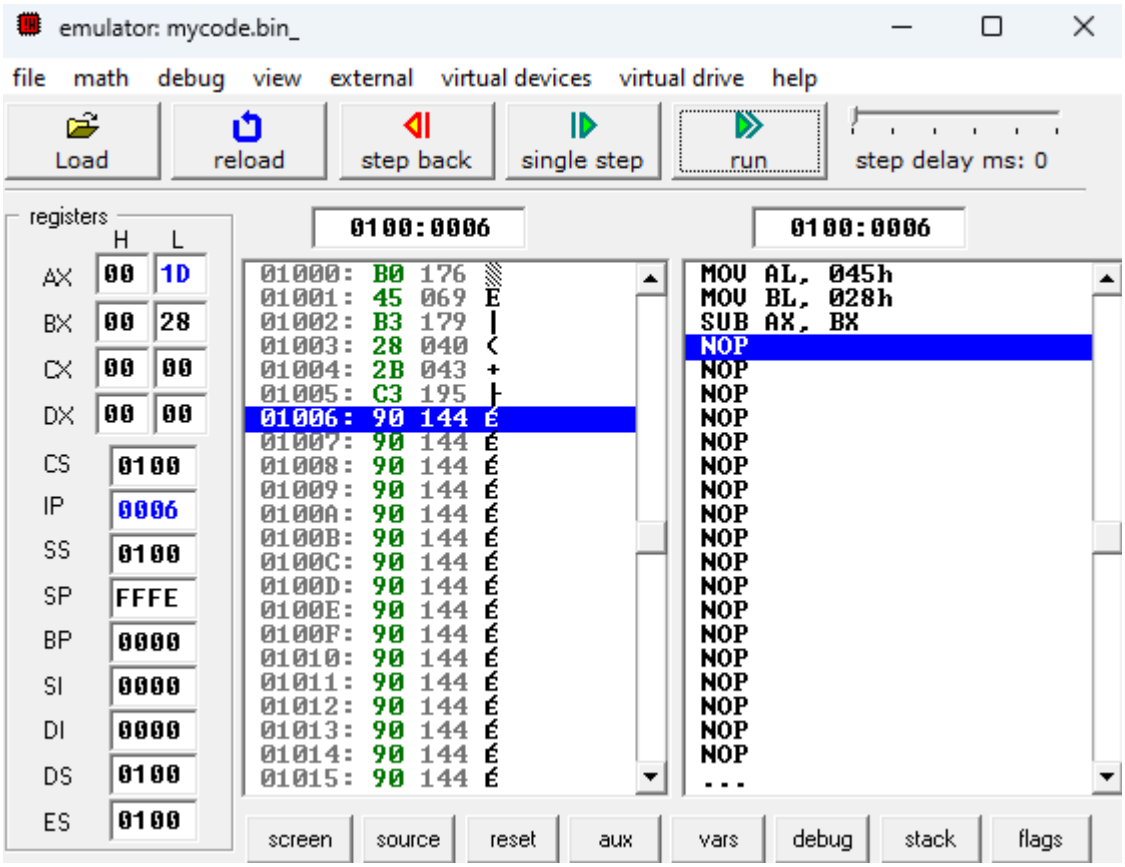
1. Load the first number in AX.
2. Load the second number. in BL
3. Divide AX by BL.
4. After division, the quotient is in AL and the remainder is in AH.

ADDITION



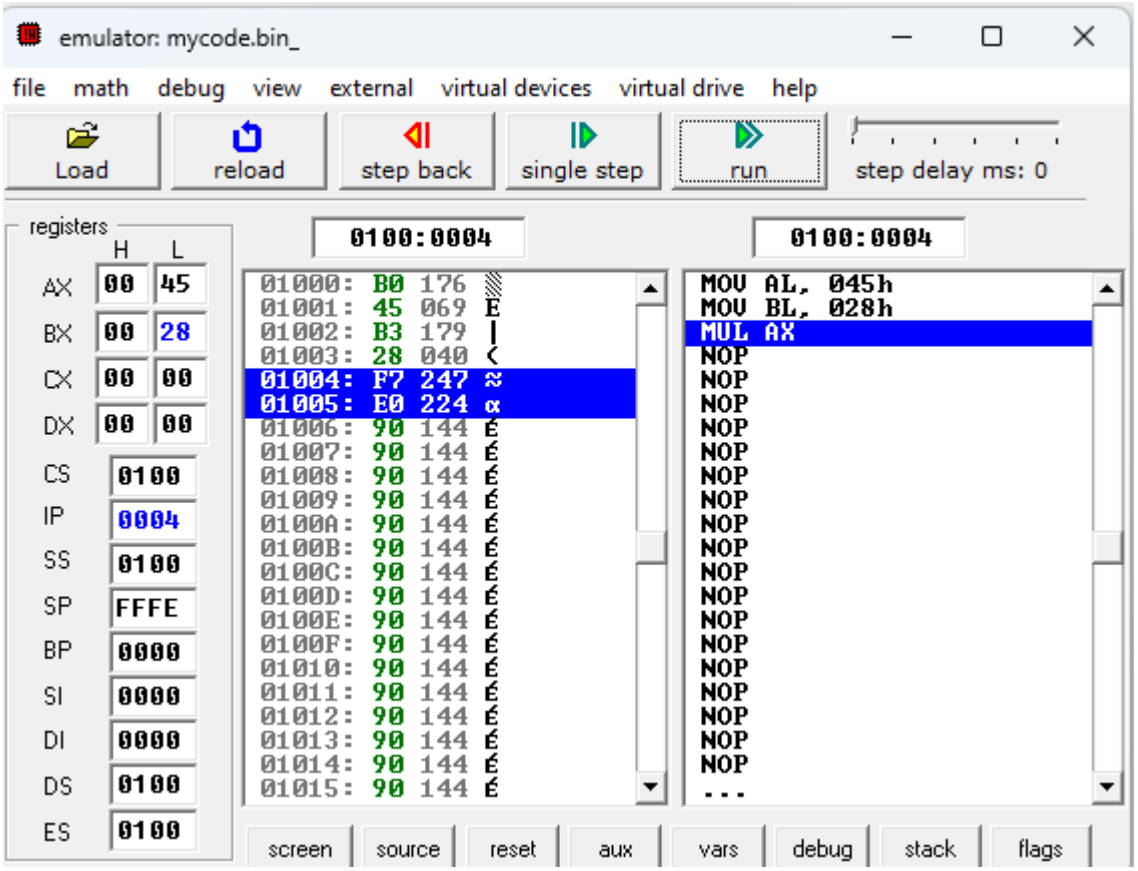
SUBSTRACTION

```
MOV AL, 45H
MOV BL, 28H
SUB AX,BX
```



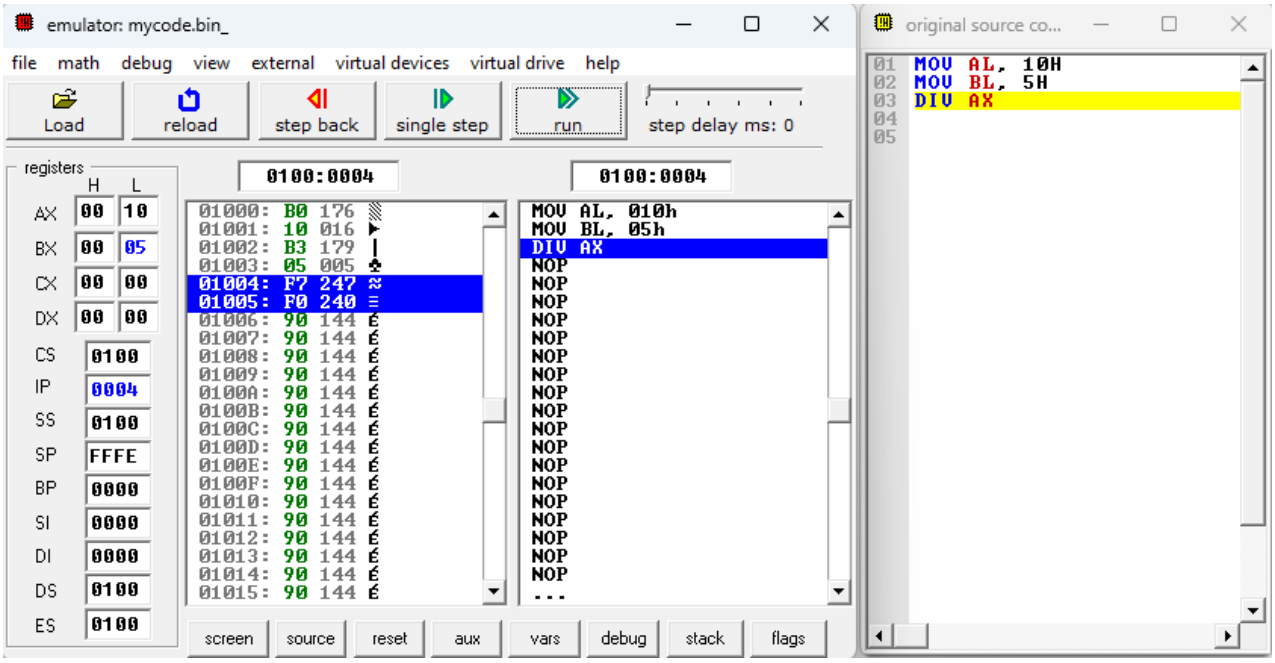
MULTIPLICATION

```
MOV AL, 45H
MOV BL, 28H
MUL AX
```



DIVISION

```
MOV AL, 45H
MOV BL, 28H
MUL AX
```


emulator: mycode.bin_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
registers
H L
AX 00 10
BX 00 05
CX 00 00
DX 00 00
CS 0100
IP 0004
SS 0100
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0100
ES 0100
0100:0004
0100:0004
01000: B0 176
01001: 10 016
01002: B3 179
01003: 05 005
01004: F7 247
01005: F0 240
MOV AL, 010h
MOV BL, 05h
DIV AX
NOP
screen source reset aux vars debug stack flags
original source co...
01 MOV AL, 10H
02 MOV BL, 5H
03 DIV AX

Conclusion:

In summary, the Intel 8086 microprocessor stands out with its 16-bit architecture, extensive addressing capabilities, and versatile instruction set. Its features include a 20-bit address bus, 16-bit data bus, multiplexed address and data bus, and various operating modes. The 8086 played a pivotal role in early computing systems, shaping the landscape of modern CPUs.

1. Explain the features of 8086.

- 16-Bit Microprocessor:

The 8086 is a 16-bit microprocessor. This means that its arithmetic logic unit (ALU), internal registers, and most instructions operate on 16-bit binary words.

- Data Bus:

The 8086 has a 16-bit data bus, allowing it to read or write data from/to memory and ports in either 16-bit or 8-bit chunks.
Its sibling, the 8088, has an 8-bit data bus, limiting it to 8-bit data transfers.

- Address Bus:

The 8086 boasts a 20-bit address bus, enabling it to directly access 1 megabyte (1MB) of memory (2^20 or 1,048,576 memory locations).
Each memory location stores a byte, so sixteen-bit words span two consecutive memory locations.
The 8088 also has a 20-bit address bus, allowing it to address the same memory range.

- I/O Ports:

The 8086 can generate 16-bit I/O addresses, granting access to 65,536 I/O ports.

- Registers:

It provides fourteen 16-bit registers, which play crucial roles in data manipulation and storage.
Multiplexed Address and Data Bus:
The 8086 employs a multiplexed address and data bus, reducing the number of pins needed.
However, this design choice slightly slows down data transfer.

- Clock Rates:

The 8086 requires a single-phase clock with a 33% duty cycle for optimized internal timing.
Clock rates for different variants include:
5 MHz for the standard 8086
8 MHz for the 8086-2
10 MHz for the 8086-1

- Instruction Set:

The 8086 provides a powerful instruction set with various addressing modes:
Register
Immediate
Direct
Indirect through an index or base
Indirect through the sum of a base and an index register
Relative
Implied

- Operating Modes:

The 8086 operates in two modes:

Minimum mode: Used when only one 8086 CPU is in a microcomputer system.
Maximum mode: Used in multiprocessor systems with external bus controllers (e.g., 8288).
Multiprogramming Support:
The 8086 supports multiprogramming, allowing multiple processes to execute in a time-multiplexed fashion.

- Instruction Queue:

The 8086 fetches up to six instruction bytes (four for the 8088) from memory and queues them for faster execution.

2. Explain general purpose and special purpose registers.

General Purpose Registers:

These registers serve as software scratchpads within the CPU.They are used during computations to:

- Pass parameters to functions.
- Store return values.
- Hold intermediate values during operations.

General purpose registers are versatile and can be employed for various purposes.

Examples of general purpose registers include:

- Accumulator (AX): Used for arithmetic, logic, and data transfer instructions. It can execute both 8-bit and 16-bit instructions.
- Base (BX): Often used as a data pointer for indirect addressing.
- Count (CX): Primarily used for loop control and string operations.

Special Purpose Registers:

These registers are designated solely for specific tasks and have predefined roles.They hold critical information related to program execution and system state.

Examples of special purpose registers include:

- Program Counter (PC): Keeps track of the memory address of the next instruction to be executed.
- Stack Pointer (SP): Manages the stack, which is crucial for function calls and local variables.
- Segment Registers (CS, DS, SS, ES, FS, GS): Control memory segmentation and address translation.