



Vidyavardhini's College of Engineering & Technology  
Department of Artificial Intelligence and Data Science (AI&DS)

---

<b>Name:</b>	Aniruddh Sawant
<b>Roll No:</b>	52
<b>Class/Sem:</b>	SE/IV
<b>Experiment No.:</b>	1
<b>Title:</b>	To perform basic arithmetic operations on 16-bit data.
<b>Date of Performance:</b>	23/01/2024
<b>Date of Submission:</b>	30/01/2024
<b>Marks:</b>	
<b>Sign of Faculty:</b>	

**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:



## Vidyavardhini's College of Engineering & Technology

### Department of Artificial Intelligence and Data Science (AI&DS)

---

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.



# Vidyavardhini's College of Engineering & Technology

## Department of Artificial Intelligence and Data Science (AI&DS)

### 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

The screenshot displays an 8086 emulator interface. On the left, the 'registers' window shows the state of various registers: AX (0006), BX (0002), CX (0000), DX (0000), CS (0100), IP (0008), SS (0100), SP (FFFE), BP (0000), SI (0000), DI (0000), DS (0100), and ES (0100). The 'instruction' window shows the current instruction being executed: 'ADD AX, BX'. The 'source code' window shows the assembly code being executed: 'MOV AX, 0004h', 'MOV BX, 0002h', and 'ADD AX, BX'. The 'debug' window shows the memory contents at the current instruction pointer (0008), which is 0006.



## Subtraction

MOV AX, 0004H  
MOV

BX, 0002H

SUB AX, BX

emulator: noname.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	02
BX	00	02
CX	00	00
DX	00	00
CS	0100	
IP	0008	
SS	0100	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0100	
ES	0100	

0100:0008

Address	Hex	ASCII
010000	B8 184	
010001	04 004	
010002	00 000	NULL
010003	BB 187	
010004	02 002	
010005	00 000	NULL
010006	2B 043	
010007	C3 195	
010008	90 144	E
010009	90 144	E
01000A	90 144	E
01000B	90 144	E
01000C	90 144	E
01000D	90 144	E
01000E	90 144	E
01000F	90 144	E
010010	90 144	E
010011	90 144	E
010012	90 144	E
010013	90 144	E
010014	90 144	E
010015	90 144	E

0100:0008

```
MOV AX, 00004h
MOV BX, 00002h
SUB AX, BX
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

## Multiplication

MOV AX, 0004H  
MOV

BX, 0002H

MUL BX

emulator: noname.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	08
BX	00	02
CX	00	00
DX	00	00
CS	0100	
IP	0008	
SS	0100	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0100	
ES	0100	

0100:0008

Address	Hex	ASCII
010000	B8 184	
010001	04 004	
010002	00 000	NULL
010003	BB 187	
010004	02 002	
010005	00 000	NULL
010006	F7 247	
010007	E3 227	
010008	90 144	E
010009	90 144	E
01000A	90 144	E
01000B	90 144	E
01000C	90 144	E
01000D	90 144	E
01000E	90 144	E
01000F	90 144	E
010010	90 144	E
010011	90 144	E
010012	90 144	E
010013	90 144	E
010014	90 144	E
010015	90 144	E

0100:0008

```
MOV AX, 00004h
MOV BX, 00002h
MUL BX
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```



# Vidyavardhini's College of Engineering & Technology

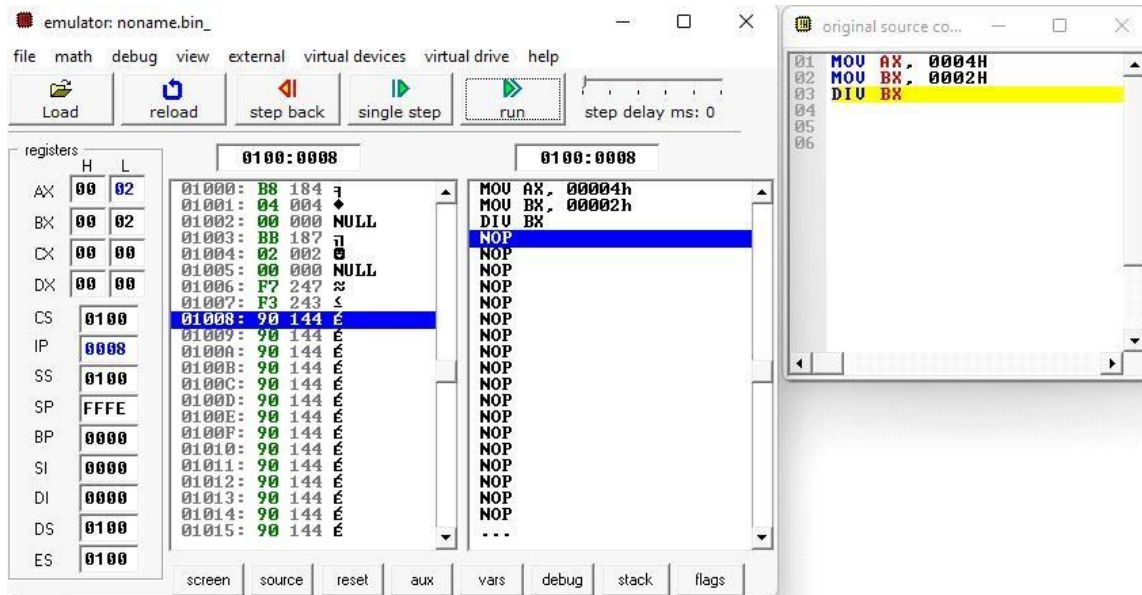
## Department of Artificial Intelligence and Data Science (AI&DS)

### 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.



## Vidyavardhini's College of Engineering & Technology

### Department of Artificial Intelligence and Data Science (AI&DS)

---

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.



---

## ~~Vidyavardhini's College of Engineering & Technology~~

### Department of Artificial Intelligence and Data Science (AI&DS)

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.