

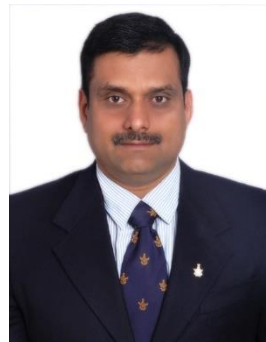


Microprocessors and Interfaces: 2021-22

Lab 2

8086 Arithmetic Operations

By Dr. Sanjay Vidhyadharan



- 2.1 Addition of two numbers

- Instruction to be used: **ADD**

Instruction	Operands	Usage Description
ADD	REG, memory	Operand 1 = Operand 1 + Operand 2 Example: ADD AL,BL ADD AL,-5 Operates on both 8-bit or 16-bit numbers.
	memory, REG	
	REG, REG	
	memory, immediate	
	REG, immediate	

C	Z	S	O	P	A
r	r	r	r	r	r

r: flag value depends on result of the instruction

- Effect on Flags

• 2.1 Addition of two 8-bit numbers

• Example Code

- `org 100h`
- `MOV AL, 0F0h`
- `MOV BL, 010h`
- `ADD AL, BL`
- `ret`

1. Click here and check the changes in the registers.

2. Click here to check the conditions of flags of the μ p.

3. Interpret the results of the flags of the μ p.

CF: Carry is generated when performing n bit operations and the result is more than n bits, then it is 1, otherwise 0.

ZF: After any arithmetical or logical operation results 0 (00)H, the zero flag is 1, otherwise 0.

PF: 1 accumulator has even number of 1 bits
0 accumulator has odd parity

• 2.2 Addition of two 16-bit numbers

• Example Code

- `org 100h`
- `MOV CX, 1234H`
- `MOV DX, 5678H`
- `ADD CX, DX`
- `ret`

emulator: noname.com_

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	00
BX	00	00
CX	68	AC
DX	56	78
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154 F400:0154

1. Click here and check the changes in the registers.

```
F4157: 00 000 NULL ADD [BX + SI], AL
F4158: 00 000 NULL ADD BH, BH
F4159: 00 000 NULL DEC BP
F415A: 00 000 NULL SBB CL, BH
F415B: 00 000 NULL ADD [BX + SI], AL
F415C: 00 000 NULL ADD [BX + SI], AL
F415D: 00 000 NULL ADD [BX + SI], AL
F415E: 00 000 NULL ADD [BX + SI], AL
F415F: 00 000 NULL ADD [BX + SI], AL
F4160: FF 255 RES ADD BH, BH
F4161: FF 255 RES DEC BP
F4162: CD 205 = ADD BH, CL
F4163: 1A 026 → ADD [BX + SI], AL
F4164: CF 207 ± ADD [BX + SI], AL
F4165: 00 000 NULL ...
```

screen source reset aux vars debug stack flags

flags

CF 0
ZF 0
SF 0
OF 0
PF 1
AF 0
IF 0
DF 0

analyse

2. Click here to check the conditions of flags of the μp.

3. Interpret the results of the flags of the μp.

• 2.3 Subtraction of two numbers

• Instruction to be used: **SUB**

Instruction	Operands	Usage Description
SUB	REG, memory	Operand 1 = Operand 1 - Operand 2 Example: SUB AL,BL SUB AL,1 Operates on both 8-bit or 16-bit numbers.
	memory, REG	
	REG, REG	
	memory, immediate	
	REG, immediate	

C	Z	S	O	P	A
r	r	r	r	r	r

r: flag value depends on result of the instruction

• Effect on Flags

2.3 Subtraction of two 8-bit numbers

Example Code

- `org 100h`
- `MOV AL, 009H`
- `MOV BL, 006H`
- `SUB AL, BL`
- `ret`

emulator: noname.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

AX	00 03
BX	00 06
CX	00 07
DX	00 00
CS	F400
IP	0154
SS	0700
SP	FFFA
BP	0000
SI	0000
DI	0000
DS	0700
ES	0700

F400:0154

F4150: FF 255 RES

BIOS DI

1. Click here and check the changes in the registers.

flags

CF	0
ZF	0
SF	0
OF	0
PF	1
AF	0
IF	0
DF	0

analyse

screen source reset aux vars debug stack flags

2. Click here to check the conditions of flags of the μ p.

3. Interpret the results of the flags of the μ p.

• 2.4 Subtraction of two 16-bit numbers

• Example Code

- `org 100h`
- `MOV AX, 0FCBAH`
- `MOV BX, 01D3FH`
- `SUB AX, BX`
- `ret`

emulator: noname.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

AX	DF	7B
BX	1D	3F
CX	00	09
DX	00	00
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154 F400:0154

F4150: FF 255 RES BIOS DI
F4151: FF 255 RES INT 0201
F4152: 00 000 NULL
F4153: 00 000 NULL
F4154: 00 000 NULL
F4155: 00 000 NULL
F4156: 00 000 NULL
F4157: 00 000 NULL
F4158: 00 000 NULL ADD BH, BH
F4159: 00 000 NULL DEC BP
F415A: 00 000 NULL SBB CL, BH
F415B: 00 000 NULL ADD [BX + SI], AL
F415C: 00 000 NULL ADD [BX + SI], AL
F415D: 00 000 NULL ADD [BX + SI], AL
F415E: 00 000 NULL ADD [BX + SI], AL
F415F: 00 000 NULL ADD [BX + SI], AL
F4160: FF 255 RES ADD BH, BH
F4161: FF 255 RES DEC BP
F4162: CD 205 = ADD BH, CL
F4163: 1A 026 → ADD [BX + SI], AL
F4164: CF 207 ± ADD [BX + SI], AL
F4165: 00 000 NULL ...

screen source reset aux vars debug stack flags

flags

CF 0
ZF 0
SF 1
OF 0
PF 1
AF 1
IF 0
DF 0

analyse

1. Click here and check the changes in the registers.

2. Click here to check the conditions of flags of the µp.

3. Interpret the results of the flags of the µp.

SF: 1- MSB is 1 (negative)
0- MSB is 0 (positive)

AF: 1-carry out from bit 3 on addition or borrow into bit 3 on subtraction
0-otherwise

• 2.5 Multiplication of two numbers

- Instruction to be used: **MUL**

Instruction	Operands	Usage Description
MUL	REG	When operand is a byte : $AX = AL \times \text{operand}$.
	memory	When operand is a word : $(DX AX) = AX \times \text{operand}$.

- Effect on Flags

C	Z	S	O	P	A
r	r	r	r	r	r

r: flag value depends on result of the instruction.

- CF = OF = 0 when high section of the result is zero.

• 2.5 Multiplication of two 8-bit numbers

• Example Code

- `org 100h`
- `MOV AX, 04H`
- `MOV BX, 05H`
- `MUL BX`
- `ret`

The screenshot shows an emulator window titled "emulator: noname.com_". The interface includes a menu bar (file, math, debug, view, external, virtual devices, virtual drive, help) and a toolbar with buttons for Load, reload, step back, single step, run, and a step delay slider (0 ms). On the left, a "registers" panel displays the state of various registers: AX (00 14), BX (00 05), CX (00 09), DX (00 00), CS (F400), IP (0154), SS (0700), SP (FFFA), BP (0000), SI (0000), DI (0000), DS (0700), and ES (0700). The main window is split into two panes. The left pane shows memory addresses from F4150 to F4165 with their corresponding hex values and comments (e.g., RES, SPA, NULL). The right pane shows the assembly code for these addresses, with the instruction at F4157, `ADD BH, BH`, highlighted in yellow. A "flags" panel on the right shows the status of various flags: CF, ZF, SF, OF, PF, AF, IF, and DF, all set to 0. An "analyse" button is located below the flags panel. A large "Sanj" watermark is visible in the bottom-left corner of the image.

• 2.6 Multiplication of two 16-bit numbers

• Example Code

- `org 100h`
- `MOV AX, 0111H`
- `MOV BX, 1212H`
- `MUL BX`
- `ret`

The screenshot shows an emulator window titled "emulator: noname.com_". The interface includes a menu bar (file, math, debug, view, external, virtual devices, virtual drive, help) and a toolbar with buttons for Load, reload, step back, single step, run, and a step delay slider set to 0 ms. On the left, a "registers" panel displays the state of various registers:

Register	H	L
AX	45	32
BX	12	12
CX	00	09
DX	00	13
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

The main window displays assembly code in two panes. The left pane shows memory addresses from F4150 to F4165 with their corresponding instructions and comments. The instruction at F4154, `CF 207 ±`, is highlighted in blue. The right pane shows the BIOS interrupt routine `INT 020h`, with the instruction `I RET` highlighted in blue. A "flags" panel on the right side of the window shows the status of various flags: CF (1), ZF (0), SF (0), OF (1), PF (0), AF (0), IF (0), and DF (0). At the bottom of the emulator window, there are buttons for "screen", "source", "reset", "aux", "vars", "debug", "stack", and "flags".

Check the content of both AX and DX registers.

- 2.7 Division of two numbers
- Instruction to be used: **DIV**

Instruction	Operands	Usage Description
DIV	REG	When operand is a byte : AL = AX / operand AH = remainder (modulus)
	memory	When operand is a word : AX = (DX AX) / operand DX = remainder (modulus)

C	Z	S	O	P	A
r	r	r	r	r	r

Effect on Flags

• 2.7 Division of two 8-bit numbers

• Example Code

- `org 100h`
- `MOV AX, 20H`
- `MOV BX, 10H`
- `DIV BX`
- `ret`

The screenshot shows an emulator window titled "emulator: noname.com_". The interface includes a menu bar (file, math, debug, view, external, virtual devices, virtual drive, help) and a toolbar with buttons for Load, reload, step back, single step, run, and a step delay slider set to 0 ms. On the left, a "registers" panel displays the state of various registers: AX (00 02), BX (00 10), CX (00 09), DX (00 00), CS (F400), IP (0154), SS (0700), SP (FFFA), BP (0000), SI (0000), DI (0000), DS (0700), and ES (0700). The main window is split into two panes. The left pane shows memory addresses from F4150 to F4165 with their corresponding hex values and ASCII characters: F4150: FF 255 RES, F4151: FF 255 RES, F4152: CD 205 =, F4153: 20 032 SPA, F4154: CF 207 ± (highlighted), F4155: 00 000 NULL, F4156: 00 000 NULL, F4157: 00 000 NULL, F4158: 00 000 NULL, F4159: 00 000 NULL, F415A: 00 000 NULL, F415B: 00 000 NULL, F415C: 00 000 NULL, F415D: 00 000 NULL, F415E: 00 000 NULL, F415F: 00 000 NULL, F4160: FF 255 RES, F4161: FF 255 RES, F4162: CD 205 =, F4163: 1A 026 →, F4164: CF 207 ±, F4165: 00 000 NULL. The right pane shows assembly code starting with "BIOS DI INT 020h" and "IRET" (highlighted), followed by several "ADD [BX + SI], AL" instructions and "DEC BP" instructions. On the right side, a "flags" panel shows status flags: CF (0), ZF (0), SF (0), OF (0), PF (0), AF (0), IF (0), and DF (0), along with an "analyse" button.

Check the content of both AL and DL registers.

• 2.8 Division of two 16-bit numbers

• Example Code

- `org 100h`
- `MOV AX, 2312H`
- `MOV BX, 1010H`
- `DIV BX`
- `ret`

The screenshot shows an emulator window titled "emulator: noname.com_". The menu bar includes "file", "math", "debug", "view", "external", "virtual devices", "virtual drive", and "help". The toolbar contains "Load", "reload", "step back", "single step", "run", and "step delay ms: 0".

The registers window shows the following values:

Register	H	L
AX	00	02
BX	10	10
CX	00	09
DX	02	F2
CS	0700	
IP	0108	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

The instruction window shows the following code:

```
MOV AX, 02312h
MOV BX, 01010h
DIV BX
RET
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

The instruction window also shows a memory dump with the following entries:

Address	Hex	Dec	Symbol
07100	B8	184	7
07101	12	018	#
07102	23	035	#
07103	BB	187	7
07104	10	016	7
07105	10	016	7
07106	F7	247	2
07107	F3	243	2
07108	C3	195	7
07109	90	144	E
0710A	90	144	E
0710B	90	144	E
0710C	90	144	E
0710D	90	144	E
0710E	90	144	E
0710F	90	144	E
07110	90	144	E
07111	90	144	E
07112	90	144	E
07113	90	144	E
07114	90	144	E
07115	90	144	E

The flags window shows the following values:

CF	0
ZF	0
SF	0
OF	0
PF	0
AF	0
IF	1
DF	0

The bottom of the window has buttons for "screen", "source", "reset", "aux", "vars", "debug", "stack", and "flags".

Check the content of both AX and DX registers.

• Exercise

- Write ALP codes for the following arithmetic operations:

Problem No.	Arithmetic Instructions	Which registers are to be used?
1	$12H + CAH$	CL, DL
2	$1A4CH + B1DEH$	AX, BX
3	$7AH - 4CH$	CL, DL
4	$3B7AH - C142H$	BX, CX
5	$1DH \times 77H$	AL, BL
6	$EF1AH \times CD50H$	AX, BX
7	$19H \div 03H$	AL, BL
8	$1927H \div 1201H$	AX, BX

- In each case interpret the results of different flags. Crosscheck your results by converting them into decimal numbers.

-
- Thankyou

Sanjay Vidhyadharan