Jump
Loop
Multiplication & Division

Multiplication Instruction

- Syntax:

  MUL source (for unsigned multiplication)

- If two bytes are multiplied, the product is a word (16 bits).
- If two words are multiplied, the product is a doubleword (32 bits).

Multiplication Instruction: BYTE form

- Multiplier in source byte
- Multiplicand in AL
- 16 bit product in AX
- The source may be a byte register, a memory byte, but ***not a constant***

```
MUL BL
MUL  MYBYTE
```

Multiplication Instruction: WORD form

- Multiplier in source word
- Multiplicand in AX
- Most significant 16 bits of product in DX
- Least significant 16 bits of product in AX
- The source may be a 16 bit register, a memory word but ***not a constant***

MUL BX

MUL MYWORD

Multiplication Instruction: Effect on Flags

- SF, ZF, AF, PF : Undefined
- CF/ OF:

Instruction	CF/OF
MUL	0, if the upper half of the result is zero
	1, otherwise

Multiplication Instruction: Examples

- AX=FFFF H, BX=FFFF H

Instruction	Decimal Product	Hex Product	DX	AX	CF/OF
MUL BX	4294836225	FFFE0001	FFFE	0001	1

- AL=80 H, BL=FF H

Instruction	Decimal Product	Hex Product	AH	AL	CF/OF
MUL BL	32640	7F80	7F	80	1

Division Instruction

- **Syntax:**
  DIV divisor (for **unsigned** Division)

- The quotient and the remainder have the same size as the divisor.
- All the flags are undefined
- **Divide Overflow**: If the quotient is too long to fit in the destination, program terminates and the system displays the message "Divide Overflow"

Division Instruction: BYTE form

- Divisor is an 8 bit register or memory byte
- 16 bit dividend is in AX
- 8 bit quotient is in AL
- 8 bit remainder is in AH

Division Instruction: WORD form

- Divisor is a 16 bit register or memory word
- 32 bit dividend is in DX:AX
- 16 bit quotient is in AX
- 16 bit remainder is in DX

Sign Extension of the Dividend

- Word division: If the dividend fits in AX only
  - For DIV, DX should be cleared
  - For IDIV, DX should be made sign extension of AX. The instruction CWD (Convert Word to Doubleword)will do the extension
- Byte division: If the dividend fits in AL only
  - For DIV, AH should be cleared
  - For IDIV, AH should be made sign extension of AL. The instruction CBW (Convert Byte to Word) will do the extension
- CBW and CWD has no operands and has no effect on flags

JUMP Instruction

Two types –

- *Unconditional Jump* – **JMP**
- *Conditional Jump*
  - *Signed Jump* – **JG/JNLE, JGE/JNL, JL/JNGE, JLE/JNG**
  - *Unsigned Jump* – **JA/JNBE, JAE/JNB, JB/JNAE, JBE/JNA**
  - *Single Flag Jump* – **JE/JZ, JNE/JNZ, JC, JNC, JO, JNO, JS, JNS, JP/JPE, JNP/JPO**

Syntax

JXXX `destination_label`

Range of Conditional Jump

`destination_label` must precede the jump instruction by no more than 126 bytes, or follow it by no more than 127 bytes.

We know that execution of ADD, SUB, AND, OR and similar instructions changes the content of the destination. In many situations, we do not want to affect destination. To aid with this, 8086 provides two instructions CMP and TEST with following syntaxes.

Syntax

```
CMP  destination, source
TEST destination, source
```

CMP is just like SUB and TEST is just like AND except that in both cases destinations are not changed and only flags are affected. CMP and TEST are limited by the same rules applicable for SUB and AND respectively.

Loop Instruction

A ***loop*** is a sequence of instructions that is repeated. When the repetition number is known in prior LOOP instruction can be used.

Syntax

```
LOOP destination_label
```

- `destination_label` must precede the LOOP instruction by no more than 126 bytes.
- The counter for LOOP is the CX register which is to be initialized with the count value.
- Execution of LOOP instruction automatically decrements CX and if CX $\neq$ 0, control is transferred to destination_label; otherwise the next instrcution after loop is executed.