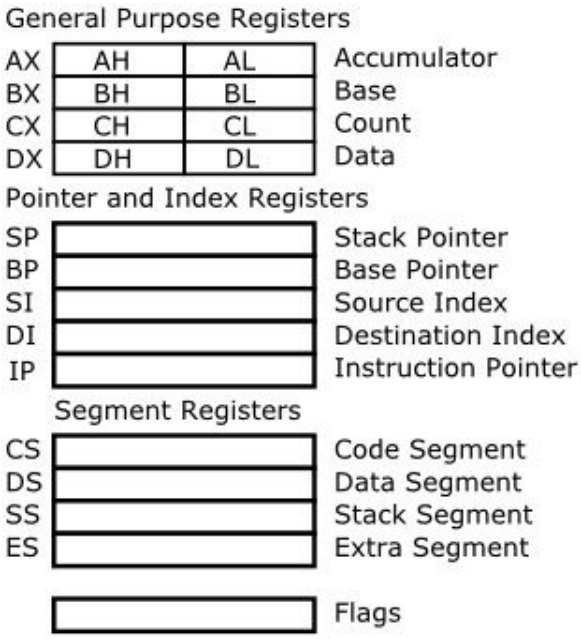
Course Code : CSE 331L / EEE 332L
Course Name : Microprocessor Interfacing & Embedded System
Faculty : Dr. Sakhawat Hussain
Instructor : Tasmia Rahman Shahidi
Class Hour : R 2:40pm - 5:50pm (section 6)
T 2:40pm - 5:50pm (section 7)
M 8:00am - 11:00am (section 8)
Semester : Spring 2021

What is Assembly Language

- A low-level programming language
- Converted to machine code using an **assembler**
- Important to low-level embedded system designs
- Designed for specific processor

Registers of MPU 8086

- Total number of registers: 14
- Each register size: 16 bits



General Purpose Registers (4 registers):

- ➢ Each GPR has two separate parts: Higher order byte and Lower order byte (each with 8 bits size). Data on each part can be separately manipulated
- ➢ Can perform 16 bits and 8 bits data read/write operations

	AH	AL
AX: 0011 0000 0011 1001 b	0011 0000 b	0011 1001 b
AX: 1111 0100 1010 0001 b	1111 0100 b	1010 0001 b
AX: F4A1 h	F4 h	A1 h
AX: 4 h	?	?

AX (Accumulator Register): Used in arithmetic, logic and data transfer operations

BX (Base Register): used as an address register

CX (Count Register): used for program loop count

DX (Data Register): used in arithmetic and I/O operations

Segment Registers (4 registers):

- ➢ Programs code, data and stack are loaded into different memory segments.
- ➢ Stack segment: used for temporary storage of addresses and data
- ➢ Code segment: program instructions are loaded in this segment.
- ➢ Data segment: variables are declared in this segment
- ➢ Extra segment: another data segment in the memory

Pointer and Index Registers (5 registers):

- ➢ Points to memory locations
- ➢ Unlike segment registers, they can be used for general arithmetic operations
- ➢ IP register: contains the offset of the next instruction in the code segment

Flag Register:

- ➢ Indicates the status of the microprocessor

Structure of Assembly Language Programming for MPU 8086

Label: OperationToPerform operand1 operand2 ;

Label: OperationToPerform Destination Source ;

Label: - symbolic name for memory location

OperationToPerform - instruction name

Operand - direct data, register, memory address

; - comments

Operands: REG, MEMORY, Immediate

- ➢ **REG:** Any valid register
- ➢ **Memory:** Referring to a memory location in RAM
- ➢ **Immediate:** Using direct values (can never be a destination)

Instruction	**Algorithm (= is assignment)**
MOV	MOV Destination, Source **Algorithm:** destination = source
ADD	ADD Destination, Source **Algorithm:** destination = destination + source
SUB	SUB Destination, Source **Algorithm:** destination = destination - source
INC	INC Destination **Algorithm:** destination = destination + 1
DEC	DEC Destination **Algorithm:** destination = destination - 1
** source remains unchanged	

```
04
05 .model small
06 .stack 100h
07 .code
08
09 mov ah, 2    ;ah=2
10 add ah, 5    ;ah=2+5=7
11 mov al, 3    ;al=3
12 sub ah, al   ;ah=ah-al=7-3=4
13
14 inc bl       ;bl = bl+1=0+1=1
15 dec dh       ;dh=dh-1=0-1=-1=ffh
16
17 mov ah, 4ch
18 int 21h
19
```