PROGRAMMING WITH 8085

Unit 2

Lecture 3

Introduction

- A microprocessor executes instructions given by the user
- Instructions should be in a language known to the microprocessor
- Microprocessor understands the language of 0’s and 1’s only
- This language is called **Machine Language**

A Machine language program to add two numbers

```
00111110                 ;Copy value 2H in
  register A
00000010
00000110                 ;Copy value 4H in
  register B
00000100
10000000                 ;A = A + B
```

Assembly Language of 8085

- It uses English like words to convey the action/meaning called as MNEMONICS
- For e.g.

MOV	to indicate data transfer
ADD	to add two values
SUB	to subtract two values

Assembly language program to add two numbers

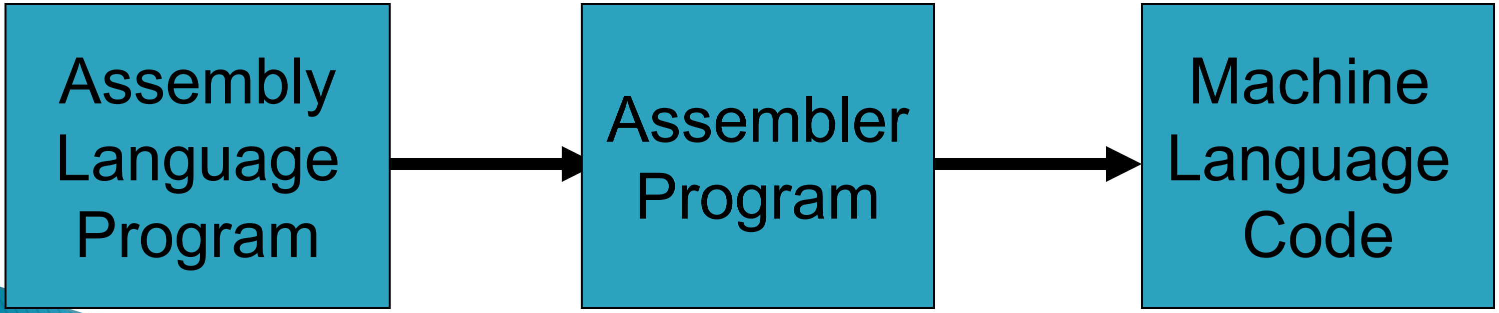
```
MVI A, 2H ;Copy value 2H in register A
MVI B, 4H ;Copy value 4H in register B
ADD B                   ;A = A + B
```

Note:

- Assembly language is specific to a given processor
- For e.g. assembly language of 8085 is different than that of Motorola 6800 microprocessor

Microprocessor understands Machine Language only

- Microprocessor cannot understand a program written in Assembly language
- A program known as **Assembler** is used to convert a Assembly language program to machine language



Assembly Language Program

Assembler Program

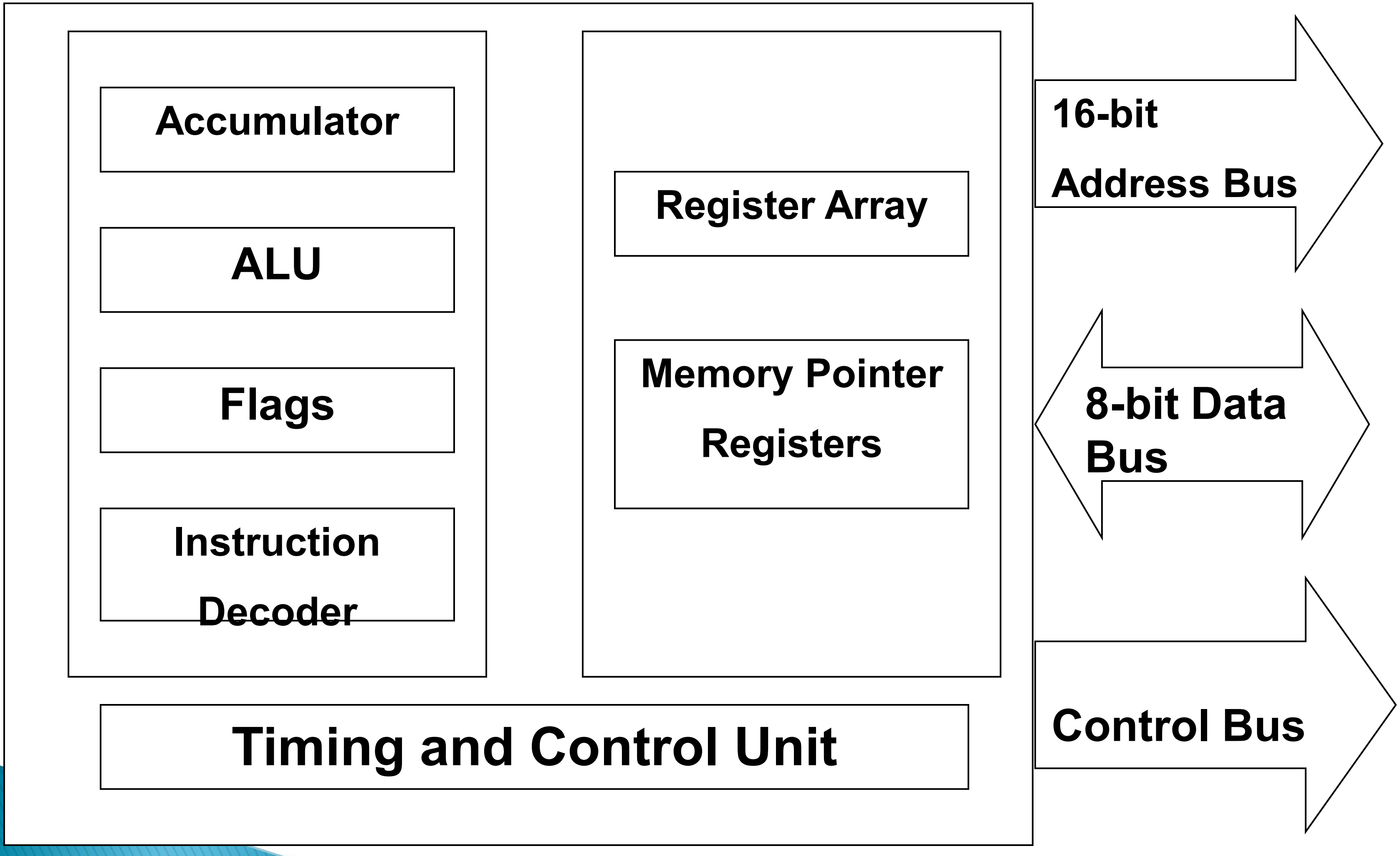
Machine Language Code

Low-level/High-level languages

- Machine language and Assembly language are both
  - Microprocessor specific (**Machine dependent**)

    so they are called
  - Low-level languages
- **Machine independent** languages are called
  - High-level languages
  - For e.g. BASIC, PASCAL,C++,C,JAVA, etc.
  - A software called **Compiler** is required to convert a high-level language program to machine code

Programming model of 8085



Accumulator

ALU

Flags

Instruction Decoder

Register Array

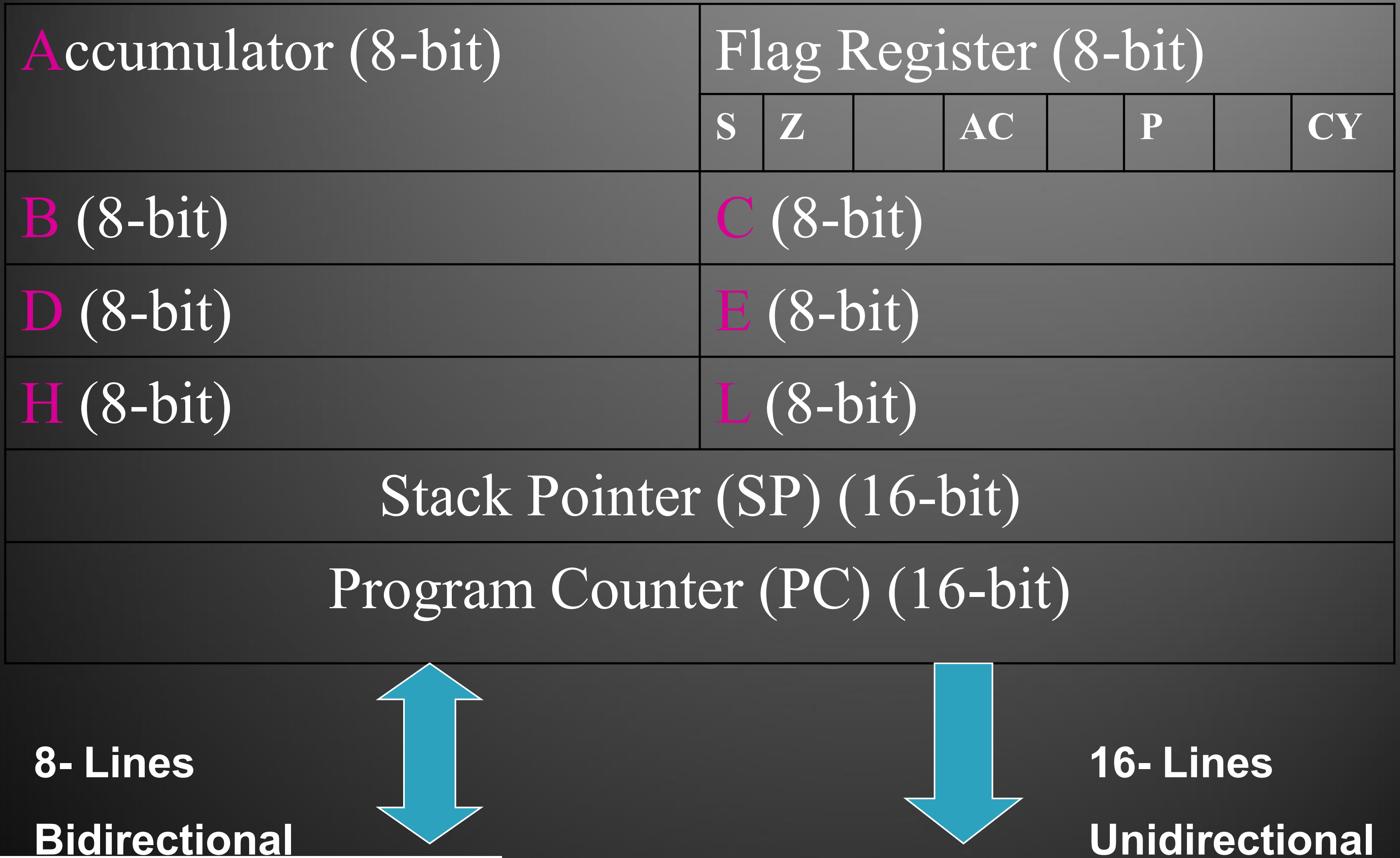
Memory Pointer Registers

Timing and Control Unit

16-bit Address Bus

8-bit Data Bus

Control Bus



<table>
<tr><td rowspan="2">Accumulator (8-bit)</td><td colspan="8">Flag Register (8-bit)</td></tr>
<tr><td>S</td><td>Z</td><td></td><td>AC</td><td></td><td>P</td><td></td><td>CY</td></tr>
<tr><td>B (8-bit)</td><td colspan="8">C (8-bit)</td></tr>
<tr><td>D (8-bit)</td><td colspan="8">E (8-bit)</td></tr>
<tr><td>H (8-bit)</td><td colspan="8">L (8-bit)</td></tr>
<tr><td colspan="9">Stack Pointer (SP) (16-bit)</td></tr>
<tr><td colspan="9">Program Counter (PC) (16-bit)</td></tr>
</table>

Overview: 8085 Programming model

1. Six general-purpose Registers
2. Accumulator Register
3. Flag Register
4. Program Counter Register
5. Stack Pointer Register

1. **Six general-purpose registers**
    - B, C, D, E, H, L
    - Can be combined as register pairs to perform 16-bit operations (BC, DE, HL)
2. **Accumulator – identified by name A**
    - This register is a part of ALU
    - 8-bit data storage
    - Performs arithmetic and logical operations
    - Result of an operation is stored in accumulator

3. **Flag Register**
   - This is also a part of ALU
   - 8085 has five flags named
     - **Zero** flag (Z)
     - **Carry** flag (CY)
     - **Sign** flag (S)
     - **Parity** flag (P)
     - **Auxiliary Carry** flag (AC)

- These flags are five flip-flops in flag register
- Execution of an arithmetic/logic operation can **set** or **reset** these flags
- Condition of flags (set or reset) can be tested through software instructions
- 8085 uses these flags in decision-making process

4. **Program Counter (PC)**
   - A 16-bit memory pointer register
   - Used to sequence execution of program instructions
   - Stores address of a memory location
     - where next instruction byte is to be fetched by the 8085
   - when 8085 gets busy to fetch current instruction from memory
     - PC is incremented by one
     - PC is now pointing to the address of next instruction

5. **Stack Pointer Register**
    - a 16-bit memory pointer register
    - Points to a location in **Stack** memory
    - Beginning of the stack is defined by loading a 16-bit address in stack pointer register

Instruction Set of 8085

- Consists of
  - 74 operation codes, e.g. MOV
  - 246 Instructions, e.g. MOV A,B
- 8085 instructions can be classified as

1. **Data Transfer (Copy)**
2. **Arithmetic**
3. **Logical and Bit manipulation**
4. **Branch**
5. **Machine Control**

Data Transfer (Copy) Operations

Copying data from a source to destination refers to data transfer function.

1. **Load** a 8-bit number in a Register
2. **Copy** from Register to Register
3. **Copy** between Register and Memory
4. **Copy** between Input / Output Port and Accumulator
5. **Load** a 16-bit number in a Register pair
6. **Copy** between Register pair and Stack memory

Example Data Transfer (Copy) Operations / Instructions

1. Load a 8-bit number 4F in register B	MVI B, 4FH
2. Copy from Register B to Register A	MOV A,B
3. Load a 16-bit number 2050 in Register pair HL	LXI H, 2050H
4. Copy from Register B to Memory Address 2050	MOV M,B
5. Copy between Input/Output Port and Accumulator	OUT 01H IN 07H

Data Transfer (Copy) Operations

6. 1 byte instruction. Processor stops executing and enters wait state.
7. 1 byte instruction . No operation .Generally used to increase processing time or substitute in place of instruction.

HLT

NOP