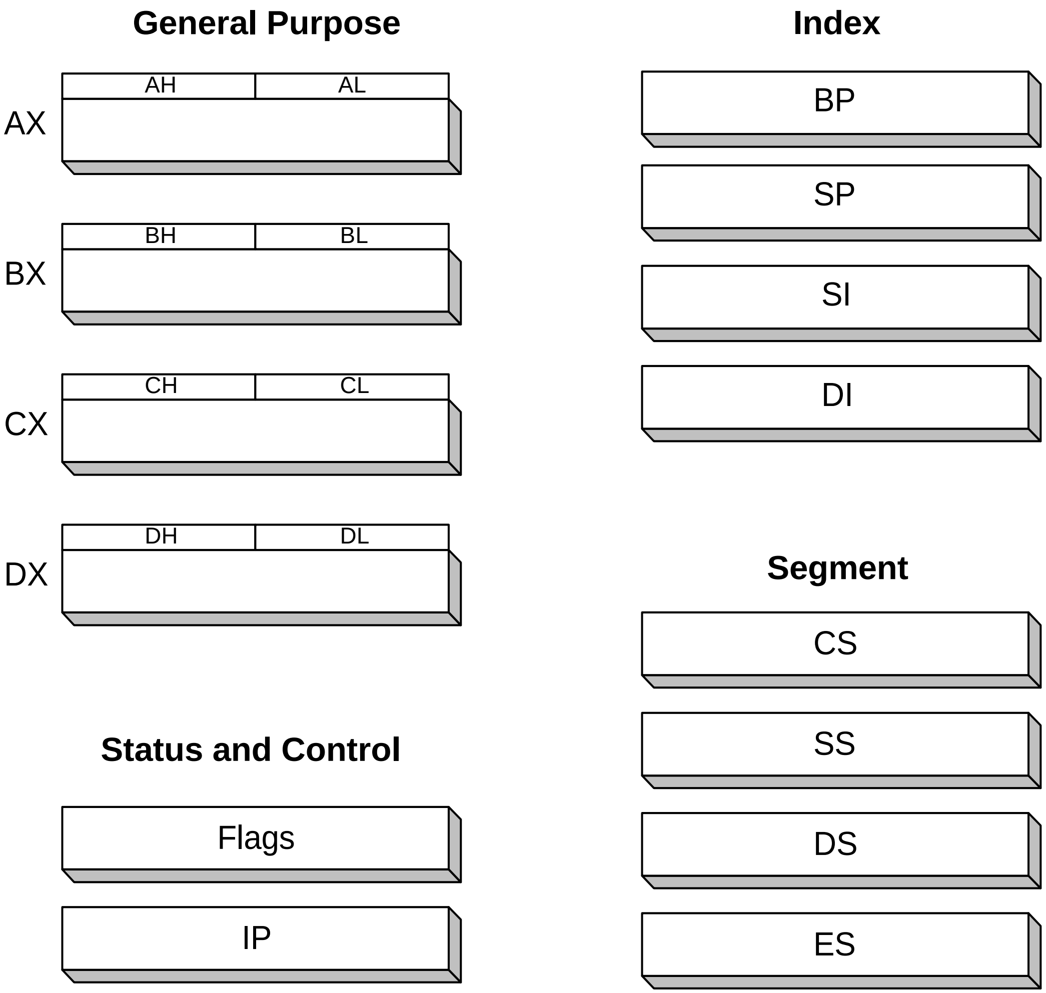
# 8086 MICROPROCESSOR

K. R Sarath Chandran, Asst
Prof/CSE

- It is a 16 bit µp.
- 8086 has a 20 bit address bus, can access upto $2^{20}$ memory locations ( 1 MB) .
- It provides 14, 16-bit registers.
- It has multiplexed address and data bus AD0- AD15 and A16 – A19.

- 8086 is designed to operate in two modes, Minimum and Maximum.
- It can prefetch upto 6 instruction bytes from memory and queues them in order to speed up instruction execution.
- It requires +5V power supply.
- 40 pin dual in line package.
- Cloak frequency - 5 MHz

# Intel 16-bit Registers



General Purpose

AX: AH AL

BX: BH BL

CX: CH CL

DX: DH DL

Status and Control

Flags

IP

Index

BP

SP

SI

DI

Segment

CS

SS

DS

ES

# 8086 Architecture

- **The 8086 architecture has two parts, the Bus Interface Unit (BIU) and the Execution Unit (EU).**

- **The BIU fetches instructions, reads and writes data, and computes the 20-bit address.**

- **The EU decodes and executes the instructions using the 16-bit ALU.**

- **The BIU contains the following registers:**

  **IP - the Instruction Pointer**<br>
  **CS - the Code Segment Register**<br>
  **DS - the Data Segment Register**<br>
  **SS - the Stack Segment Register**<br>
  **ES - the Extra Segment Register**

**The BIU fetches instructions using the CS and IP, written CS:IP, to generate the 20-bit address. Data is fetched using a segment register (usually the DS) and an effective address (EA) computed by the EU depending on the addressing mode.**

**The EU contains the following 16-bit registers:**

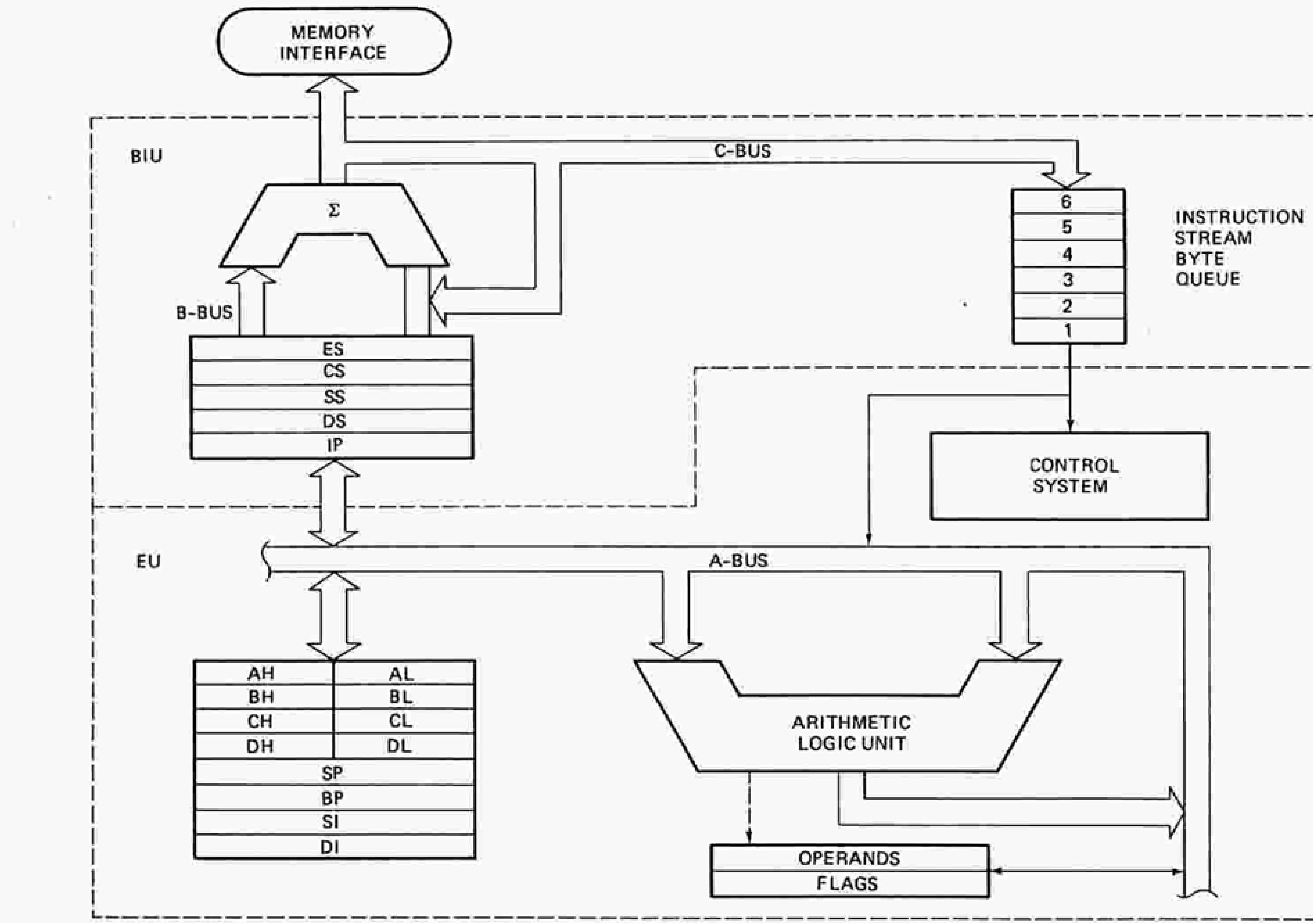
**AX - the Accumulator**
**BX - the Base Register**
**CX - the Count Register**
**DX - the Data Register**

**SP - the Stack Pointer \ defaults to stack segment**
**BP - the Base Pointer /**
**SI - the Source Index Register**
**DI - the Destination Register**

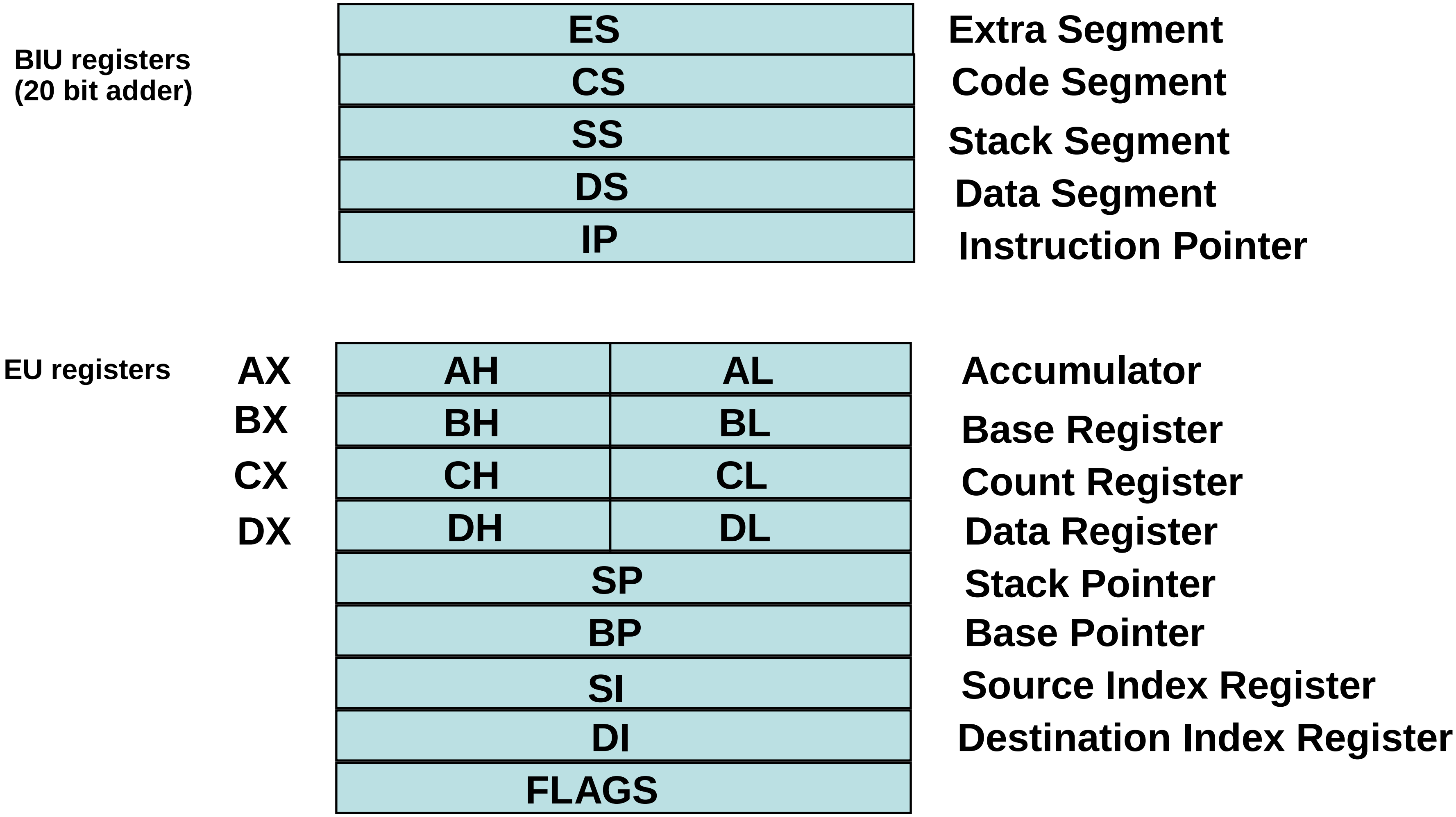
**These are referred to as general-purpose registers, although, as seen by their names, they often have a special-purpose use for some instructions.**

**The AX, BX, CX, and DX registers can be considers as two 8-bit registers, a High byte and a Low byte. This allows byte operations and compatibility with the previous generation of 8-bit processors, the 8080 and 8085. 8085 source code could be translated in 8086 code and assembled. The 8-bit registers are:**

**AX --> AH,AL**
**BX --> BH,BL**
**CX --> CH,CL**
**DX --> DH,DL**


MEMORY INTERFACE
BIU
C-BUS
Σ
B-BUS
6
5
4
3
2
1
INSTRUCTION STREAM BYTE QUEUE
ES
CS
SS
DS
IP
CONTROL SYSTEM
EU
A-BUS
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
SI
DI
ARITHMETIC LOGIC UNIT
OPERANDS
FLAGS

# 8086 Programmer’s Model



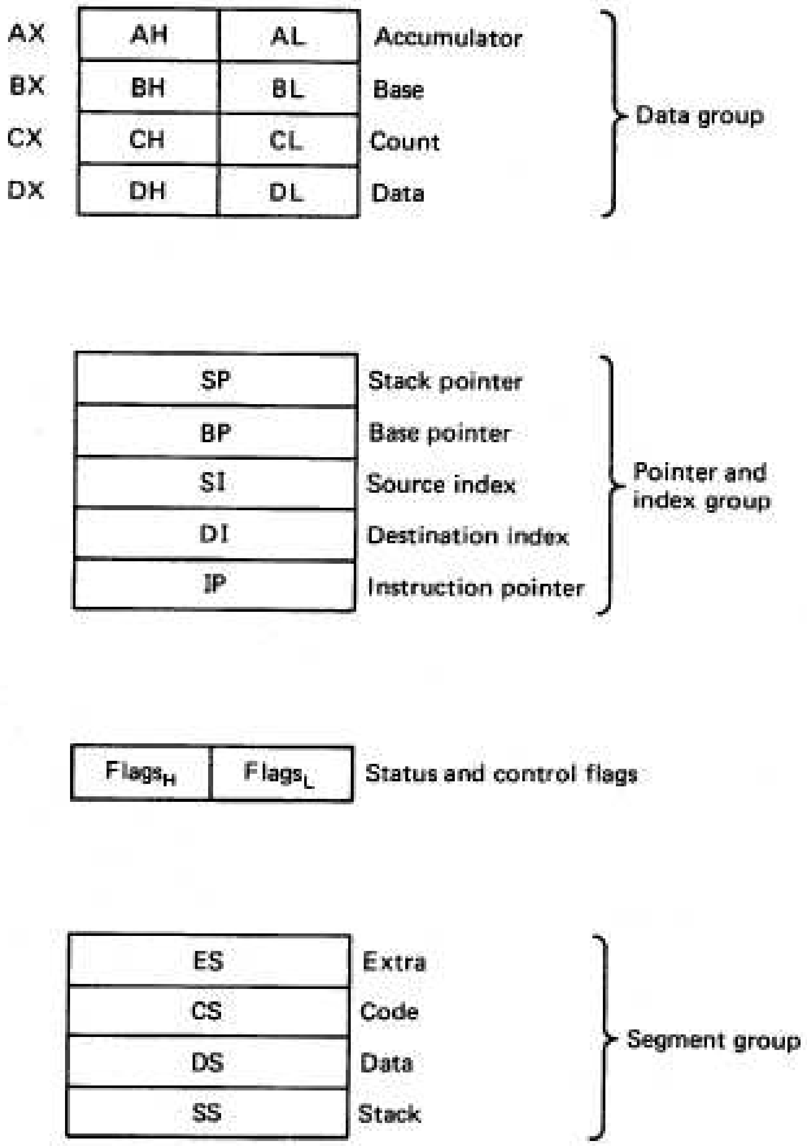
BIU registers
(20 bit adder)

| | |
|---|---|
| ES | Extra Segment |
| CS | Code Segment |
| SS | Stack Segment |
| DS | Data Segment |
| IP | Instruction Pointer |

EU registers

| | | | |
|---|---|---|---|
| AX | AH | AL | Accumulator |
| BX | BH | BL | Base Register |
| CX | CH | CL | Count Register |
| DX | DH | DL | Data Register |
| | SP | | Stack Pointer |
| | BP | | Base Pointer |
| | SI | | Source Index Register |
| | DI | | Destination Index Register |
| | FLAGS | | |

# 8086/88 internal registers 16 bits (2 bytes each)



AX | AH | AL | Accumulator
BX | BH | BL | Base
CX | CH | CL | Count
DX | DH | DL | Data

Data group

SP | Stack pointer
BP | Base pointer
SI | Source index
DI | Destination index
IP | Instruction pointer

Pointer and index group

$Flags_H$ | $Flags_L$ | Status and control flags

ES | Extra
CS | Code
DS | Data
SS | Stack

Segment group

AX, BX, CX and DX are two bytes wide and each byte can be accessed separately

These registers are used as memory *pointers*.

Flags will be discussed later

Segment registers are used as base address for a segment in the 1 M byte of memory

## The 8086/8088 Microprocessors: Registers

- Registers
  - Registers are in the CPU and are referred to by specific names
  - Data registers
    - Hold data for an operation to be performed
    - There are 4 data registers (AX, BX,  CX, DX)
  - Address registers
    - Hold the address of an instruction or data element
    - Segment registers (CS, DS, ES, SS)
    - Pointer registers (SP, BP, IP)
    - Index registers (SI, DI)
  - Status register
    - Keeps the current status of the processor

# Data Registers: AX, BX, CX, DX

- Instructions execute faster if the data is in a register
- AX, BX, CX, DX  are the data registers
- Low and High bytes of the data registers can be accessed separately
  - AH, BH, CH, DH are the high bytes
  - AL, BL, CL, and DL are the low bytes
- Data Registers are general purpose registers but they also perform special functions
- AX
  - Accumulator Register
  - Preferred register to use in arithmetic, logic and data transfer instructions

- BX
  - Base Register
  - Also serves as an address register

- CX
  - Count register
  - Used as a loop counter
  - Used in shift and rotate operations
- DX
  - Data register
  - Used in multiplication and division
  - Also used in I/O operations

## Pointer and Index Registers

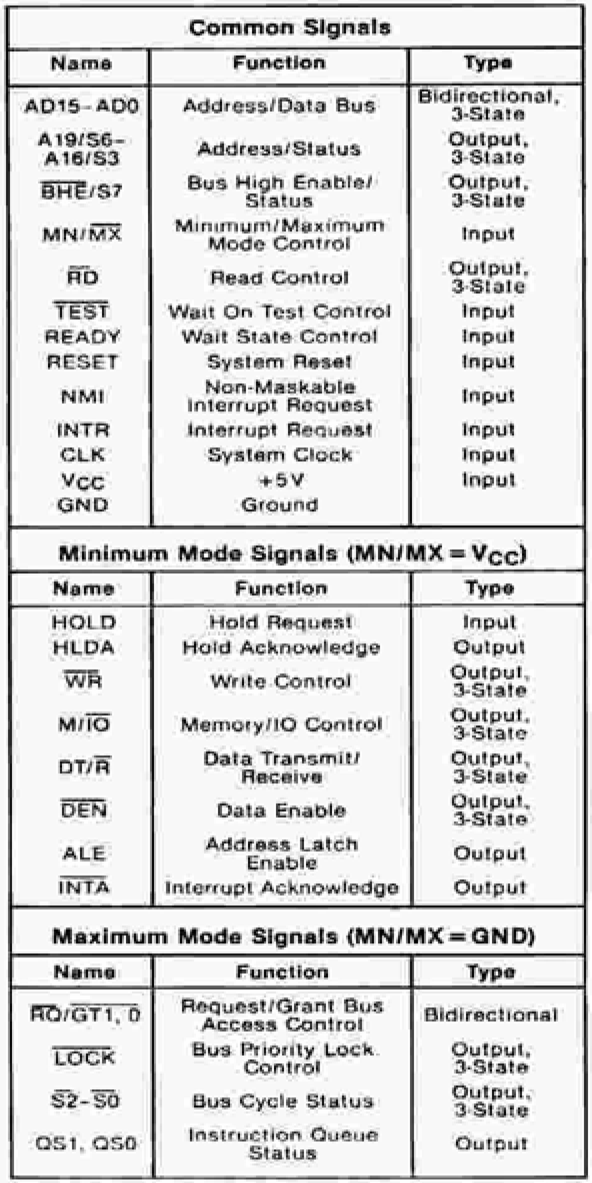
- Contain the offset addresses of memory locations
- Can also be used in arithmetic and other operations
- SP: Stack pointer
  - Used with SS to access the stack segment
- BP: Base Pointer
  - Primarily used to access data on the stack
  - Can be used to access data in other segments
- SI: Source Index register
  - is required for some string operations
  - When string operations are performed, the SI register points to memory locations in the data segment which is addressed by the DS register.  Thus, SI is associated with the DS in string operations.

- DI: Destination Index register
  - is also required for some string operations.
  - When string operations are performed, the DI register points to memory locations in the data segment which is addressed by the ES register. Thus, DI is associated with the ES in string operations.
- The SI and the DI registers may also be used to access data stored in arrays

# Segment Registers - CS, DS, SS and ES

- Are Address registers
- Store the memory addresses of instructions and data
- Memory Organization
  - Each byte in memory has a 20 bit address starting with 0 to $2^{20}$-1 or 1 meg of addressable memory
  - Addresses are expressed as 5 hex digits from 00000 - FFFFF
  - Problem: But 20 bit addresses are TOO BIG to fit in 16 bit registers!
  - Solution: Memory Segment
    - Block of 64K (65,536) consecutive memory bytes
    - A segment number is a 16 bit number
    - Segment numbers range from 0000 to FFFF
    - Within a segment, a particular memory location is specified with an offset
    - An offset also ranges from 0000 to FFFF

# 8086 Microprocessor



**Common Signals**

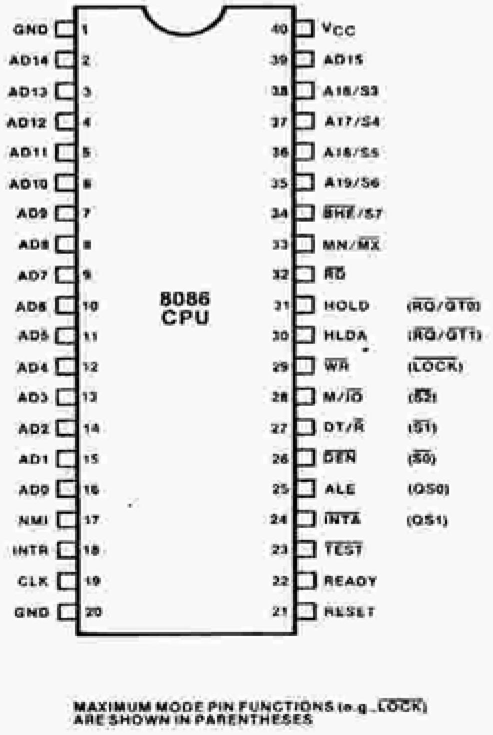
| Name | Function | Type |
|---|---|---|
| AD15–AD0 | Address/Data Bus | Bidirectional, 3-State |
| A19/S6–A16/S3 | Address/Status | Output, 3-State |
| $\overline{BHE}$/S7 | Bus High Enable/ Status | Output, 3-State |
| MN/$\overline{MX}$ | Minimum/Maximum Mode Control | Input |
| $\overline{RD}$ | Read Control | Output, 3-State |
| $\overline{TEST}$ | Wait On Test Control | Input |
| READY | Wait State Control | Input |
| RESET | System Reset | Input |
| NMI | Non-Maskable Interrupt Request | Input |
| INTR | Interrupt Request | Input |
| CLK | System Clock | Input |
| $V_{CC}$ | +5V | Input |
| GND | Ground | |

**Minimum Mode Signals (MN/MX = $V_{CC}$)**

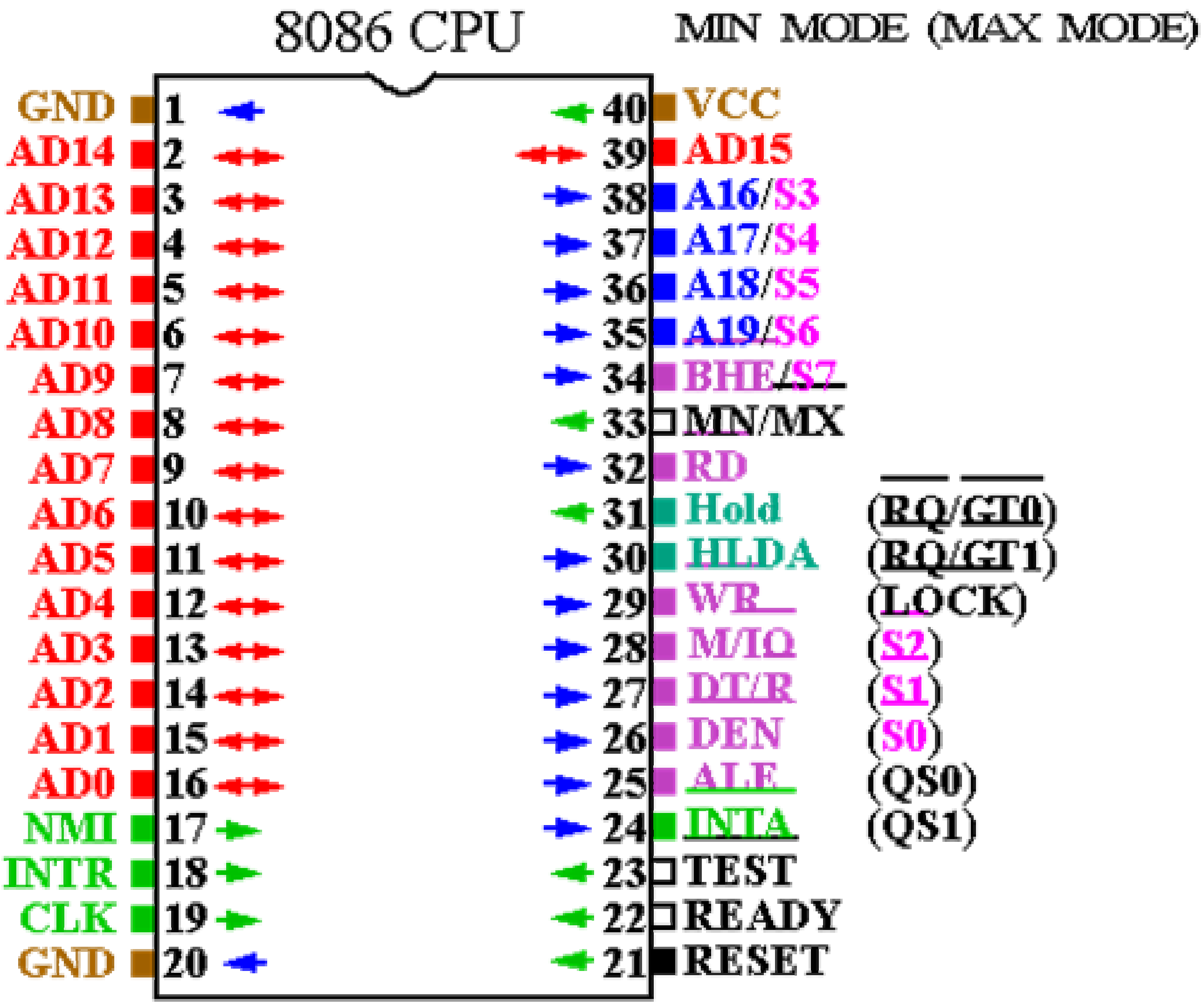
| Name | Function | Type |
|---|---|---|
| HOLD | Hold Request | Input |
| HLDA | Hold Acknowledge | Output |
| $\overline{WR}$ | Write Control | Output, 3-State |
| M/$\overline{IO}$ | Memory/IO Control | Output, 3-State |
| DT/$\overline{R}$ | Data Transmit/ Receive | Output, 3-State |
| $\overline{DEN}$ | Data Enable | Output, 3-State |
| ALE | Address Latch Enable | Output |
| $\overline{INTA}$ | Interrupt Acknowledge | Output |

**Maximum Mode Signals (MN/MX = GND)**

| Name | Function | Type |
|---|---|---|
| $\overline{RQ}$/$\overline{GT1}$, 0 | Request/Grant Bus Access Control | Bidirectional |
| $\overline{LOCK}$ | Bus Priority Lock Control | Output, 3-State |
| $\overline{S2}$–$\overline{S0}$ | Bus Cycle Status | Output, 3-State |
| QS1, QS0 | Instruction Queue Status | Output |



MAXIMUM MODE PIN FUNCTIONS (e.g. $\overline{LOCK}$) ARE SHOWN IN PARENTHESES

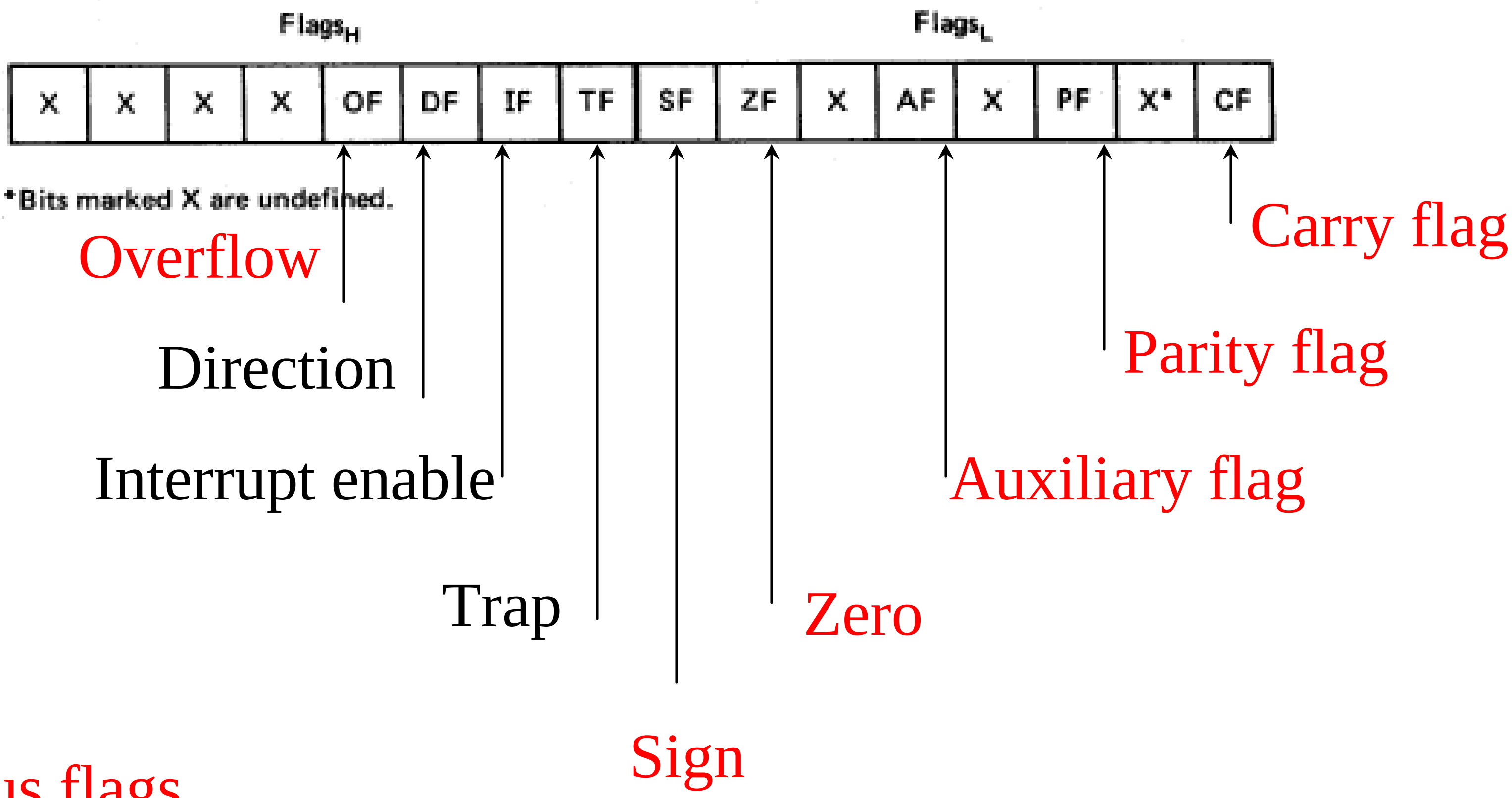


8086 CPU

MIN MODE (MAX MODE)

| | | | | |
|---|---|---|---|---|
| GND | 1 | 40 | VCC | |
| AD14 | 2 | 39 | AD15 | |
| AD13 | 3 | 38 | A16/S3 | |
| AD12 | 4 | 37 | A17/S4 | |
| AD11 | 5 | 36 | A18/S5 | |
| AD10 | 6 | 35 | A19/S6 | |
| AD9 | 7 | 34 | BHE/S7 | |
| AD8 | 8 | 33 | MN/MX | |
| AD7 | 9 | 32 | RD | |
| AD6 | 10 | 31 | Hold | (RQ/GT0) |
| AD5 | 11 | 30 | HLDA | (RQ/GT1) |
| AD4 | 12 | 29 | WR | (LOCK) |
| AD3 | 13 | 28 | M/IO | (S2) |
| AD2 | 14 | 27 | DT/R | (S1) |
| AD1 | 15 | 26 | DEN | (S0) |
| AD0 | 16 | 25 | ALE | (QS0) |
| NMI | 17 | 24 | INTA | (QS1) |
| INTR | 18 | 23 | TEST | |
| CLK | 19 | 22 | READY | |
| GND | 20 | 21 | RESET | |

# Flags



| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Flags$_H$ | | | | | | | | Flags$_L$ | | | | | | | |
| X | X | X | X | OF | DF | IF | TF | SF | ZF | X | AF | X | PF | X* | CF |

*Bits marked X are undefined.

Overflow

Direction

Interrupt enable

Trap

Sign

Zero

Auxiliary flag

Parity flag

Carry flag

6 are status flags

3 are control flag

# Flag Register

- Conditional flags:
  - They are set according to some results of arithmetic operation. You do not need to alter the value yourself.
- Control flags:
  - Used to control some operations of the MPU. These flags are to be set by you in order to achieve some specific purposes.

| Flag |  |  |  |  | O | D | I | T | S | Z |  | A |  | P |  | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit no. | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

- CF (carry) Contains carry from leftmost bit following arithmetic, also contains last bit from a shift or rotate operation.

# Flag Register

- OF (overflow) Indicates overflow of the leftmost bit during arithmetic.
- DF (direction) Indicates left or right for moving or comparing string data.
- IF (interrupt) Indicates whether external interrupts are being processed or ignored.
- TF (trap) Permits operation of the processor in single step mode.

- SF (sign) Contains the resulting sign of an arithmetic operation (1=negative)
- ZF (zero) Indicates when the result of arithmetic or a comparison is zero. (1=yes)
- AF (auxiliary carry) Contains carry out of bit 3 into bit 4 for specialized arithmetic.
- PF (parity) Indicates the number of 1 bits that result from an operation.