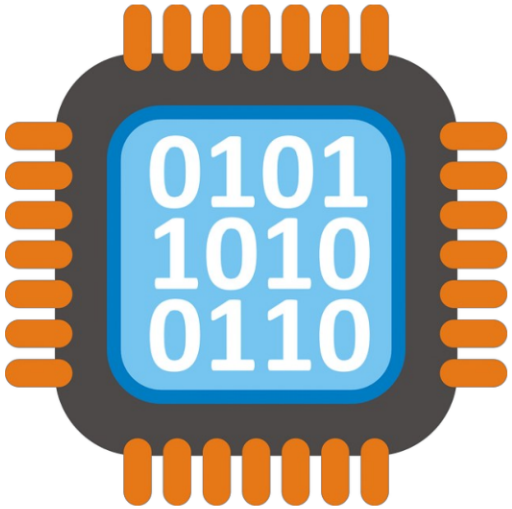
# Secure Assembly Coding

## Week # 3 Lectures




Dr. Qasem Abu Al-Haija,
*Department of Cybersecurity,*

# Intel 8086 μp

# 18086 μp Features

- 8086 is the first 16-bit **μp** released by Intel (1978).
  - 40-pin DIPs, 16-bit data bus (D0-D15) and 20-bit address bus (A0-A19).
  - Higher execution speed - larger memory size (of previous μps).
  - Run at 2.5 MIPS ➔ $T_{exe}$ of one instruction = 400 ns (=1/MIPS=1/(2.5x$10^6$)).
  - Contains a small pre-fetch 6-byte instruction queue➔ Pipelining.
  - 8086 **μp** is an example of a complex instruction set computer (CISC).
  - 8086 **μp** is an example of a von Neumann Architecture (VNA) computer.

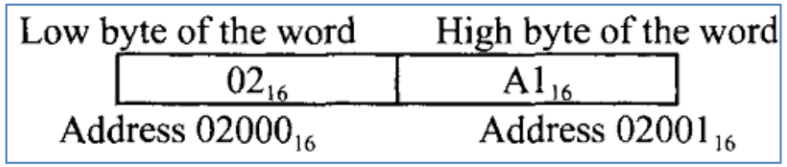
- Frequency Generation of 8086:
  - 8086 clock input signal is generated by 8284 clock generator chip.
  - Instruction execution times vary between 2 and 30 clock cycles.
  - Four versions: 8086 (5 MHz),8086- 1 (10 MHz),8086-2(8 MHz) & 8086-4 (4 MHz).

# 18086 $\mu$p Features

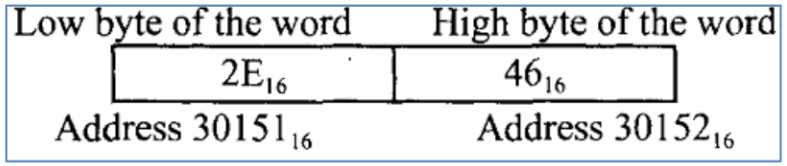
- Operational Power/ Temperature:
  - requires +5.0 V (tolerance ±10 %) and a 360 mA (max).
  - operates in ambient temperatures of between 32° F and 180° F.

- 8086 Memory Addressing:
  - 8086 has 20 address pins ➔ $2^{20}$ bytes=1 MB of memory uniquely addressable.
  - 8086 memory is Byte addressable: $00000_{16}$; $00001_{16}$; …. $FFFFF_{16}$..

- 8086 Memory Accessing:
  - 8086 has 16 data pins➔ can read 8-bit or 16-bit word (2- con. byte) from memory.
  - Word address can start at even or odd address. **Is this an issue**?
  - Ex1: The 16-bit word at address $02000_{16}$ is $A102_{16}$

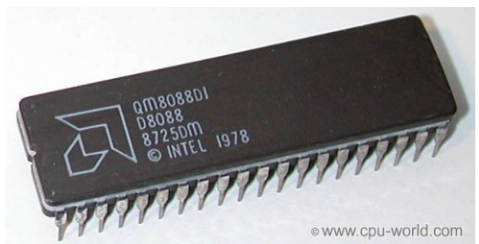



  - Ex2: The 16-bit word at address $30151_{16}$ is $462E_{16}$

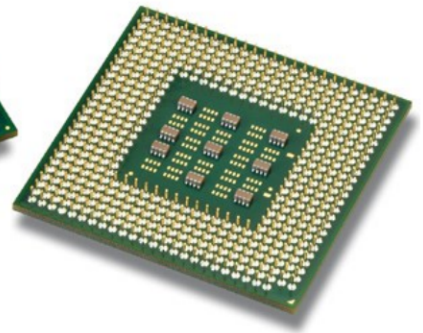


# 18086 μp Features

- 8086 Registers Naming.
  - 8086 register names followed by the letters X, H, or L (to specify 16 or 8-bit).
  - Examples: MOV AX, [START] MOV AL, [START].
- 8086 Modes of operation.
  - Uniprocessor system (minimum mode: MN/$\overline{MX}$ pin is tied to HIGH).
  - Multiprocessor system (Maximum mode: MN/$\overline{MX}$ pin is tied to LOW).
- Intel introduced many high-performance MP.
  - 80186, 80286, 80386, 80486, Pentium I, Pentium 2, Celeron, Pentium 3, Pentium 4 and others.





# 8086 Main Memory

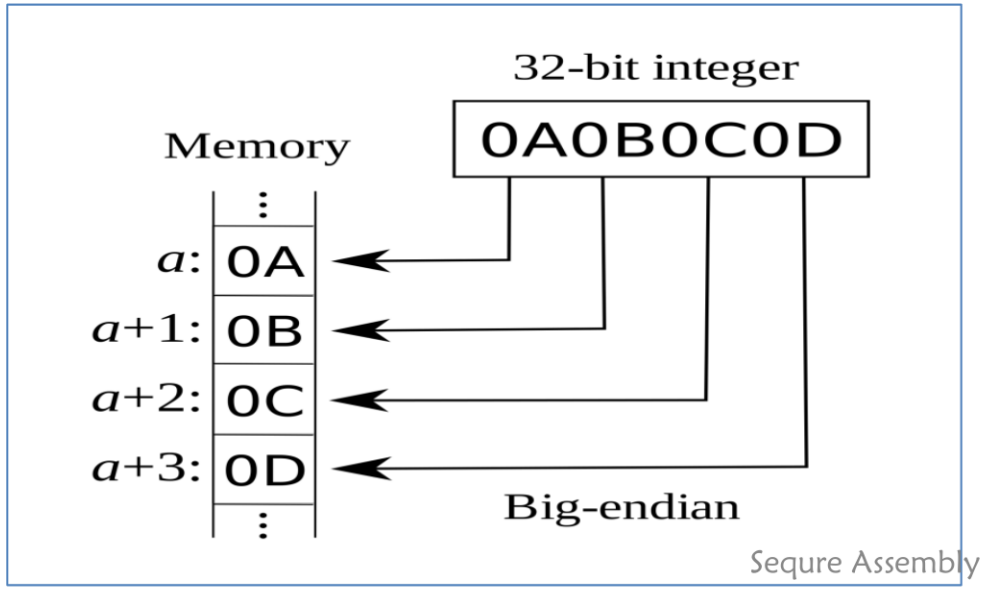
- 8086 uses a segmented memory.
  - **+Ve:** Manipulates 16-bit components only and effectively used in time-shared systems.
  - **Thus:** 8086 Memory can be divided into 16 segments (**1 MB = 16 x 64 KB** ).
  - **8086** segments may contain: codes or data or stack or extra.
  - **Therefore,** 8086 employs 16-bit registers to address segments such as: DS, CS.
  - **By this,** we will have two kind of addresses: Physical address and Logical address.
- Physical address of $\mu$P (20 bit) ➔ Not used to access Memory.
  - Instead: Logical Address with two 16-bit components [Segment: Offset] is used.
  - 8086 includes on-chip HW to translate between physical & logical addresses.
  - Shifting segment register 4 times to left then adding it to offset register.

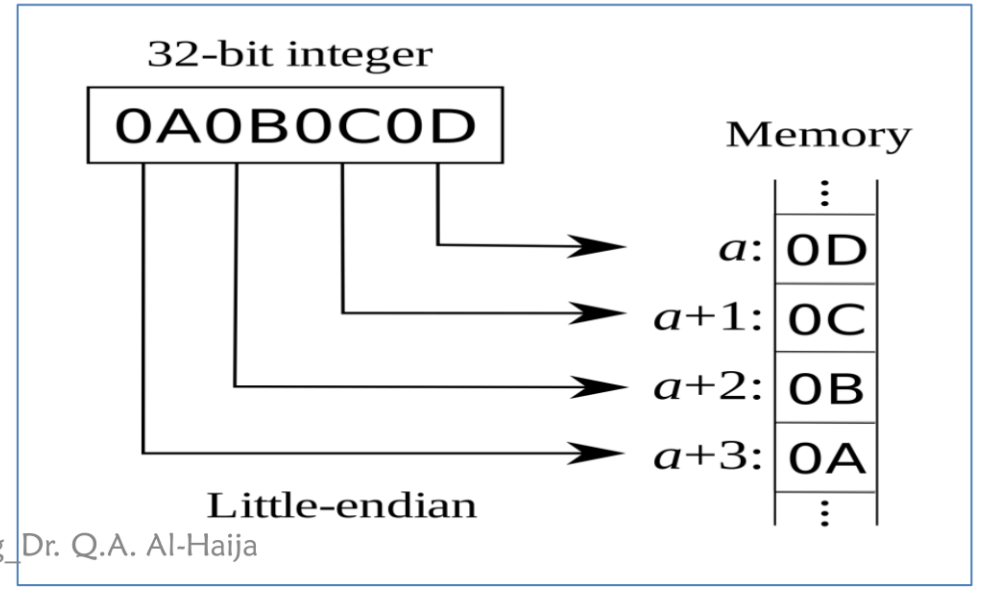
# 8086 Main Memory

- Example of address translation
  - Assume a Logical address: $2050_{16} : 0004_{16}$ ➔ [Segment : Offset].
  - We can find the physical address of 8086 $\mu$P as follows:

    1. Shift logical **4** times to left for the segment register ➔ 20500 H.
    2. Add the contents of offset register ➔ 20500H + 0004H = 20504H (Physical address)

- 8086 uses Little-endian byte ordering to compute physical address.



# 8086 Main Memory

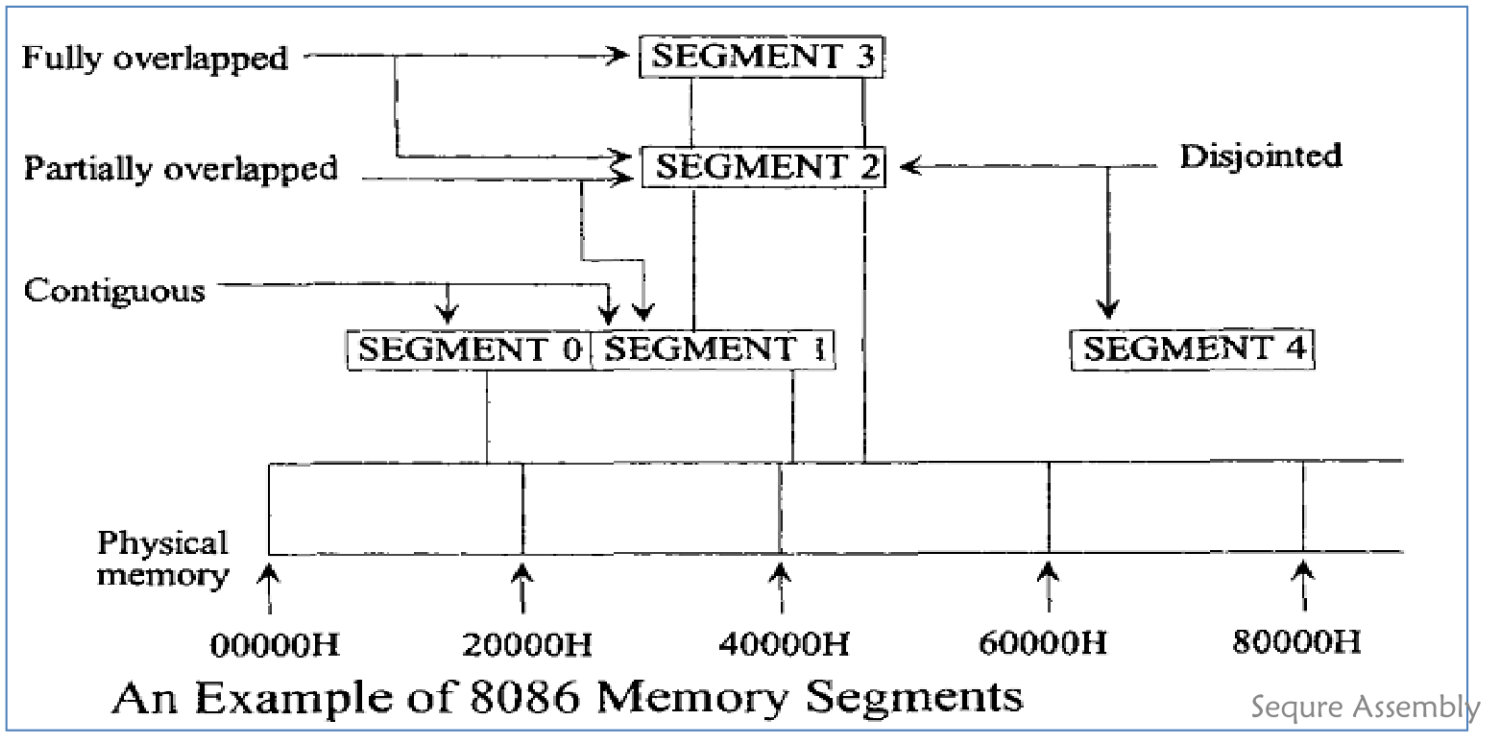
- Memory Segments Features.

– Segments can be: Contiguous, Partially overlapped, Fully overlapped, or disjointed.

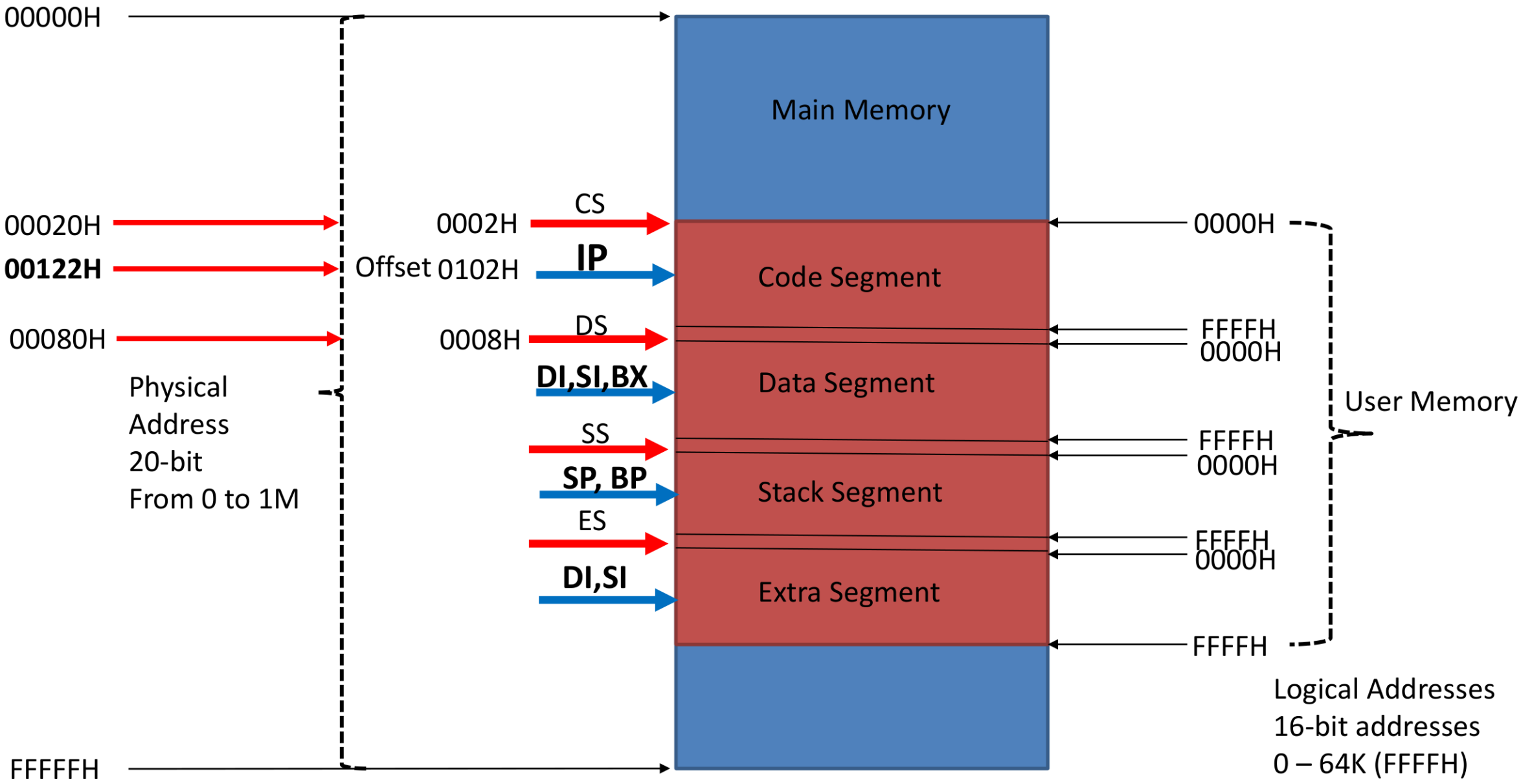
– A segment can be addressed by different segment registers (i.e. fully overlapped) .

– Every segment must start on 16-byte memory boundaries, as: 0

$0000_{16}$, $0001 0_{16}$, $00020_{16}$, $00030_{16}$ …, $FFFF0_{16}$.



An Example of 8086 Memory Segments


00000H
00020H
00122H
00080H
FFFFFH
Physical
Address
20-bit
From 0 to 1M
Main Memory
CS
0002H
IP
Offset 0102H
DS
0008H
DI,SI,BX
SS
SP, BP
ES
DI,SI
Code Segment
Data Segment
Stack Segment
Extra Segment
0000H
FFFFH
0000H
FFFFH
0000H
FFFFH
User Memory
Logical Addresses
16-bit addresses
0 – 64K (FFFFH)

# Another Example: Physical Address Calculation

- Given the CS value 0020H
- And the IP = 0121H

- What is the Physical Address?
  - Add zero to the right of the segment register, then Add it to IP
  - CS = 00200H
  - IP = 0121H
- = 00321H

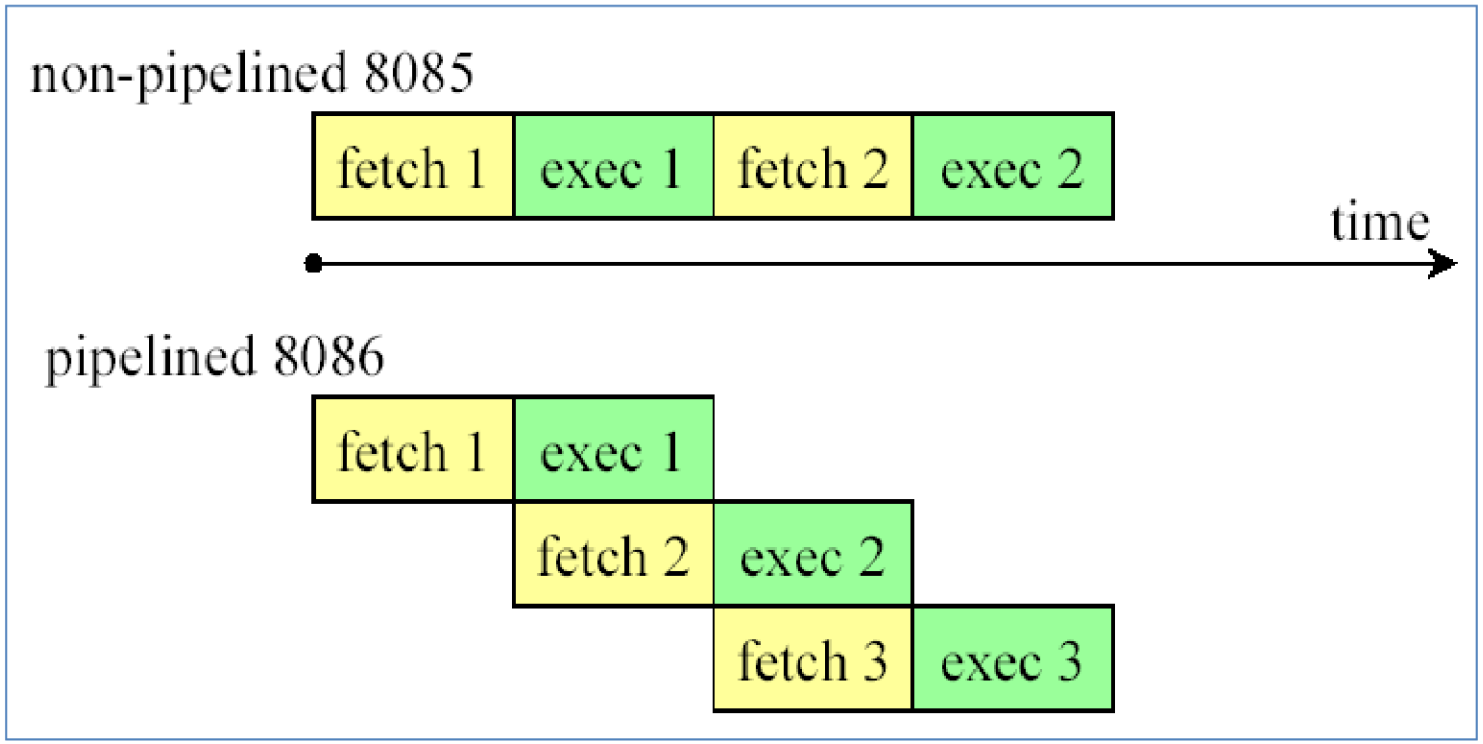
# 8086 Hardware Architecture

- Intel enhanced the internal architecture of 8086 using Pipelining.
  - Pipelining is to allow CPU to fetch and execute at the same time.
  - This can be accomplished by having several units works simultaneously

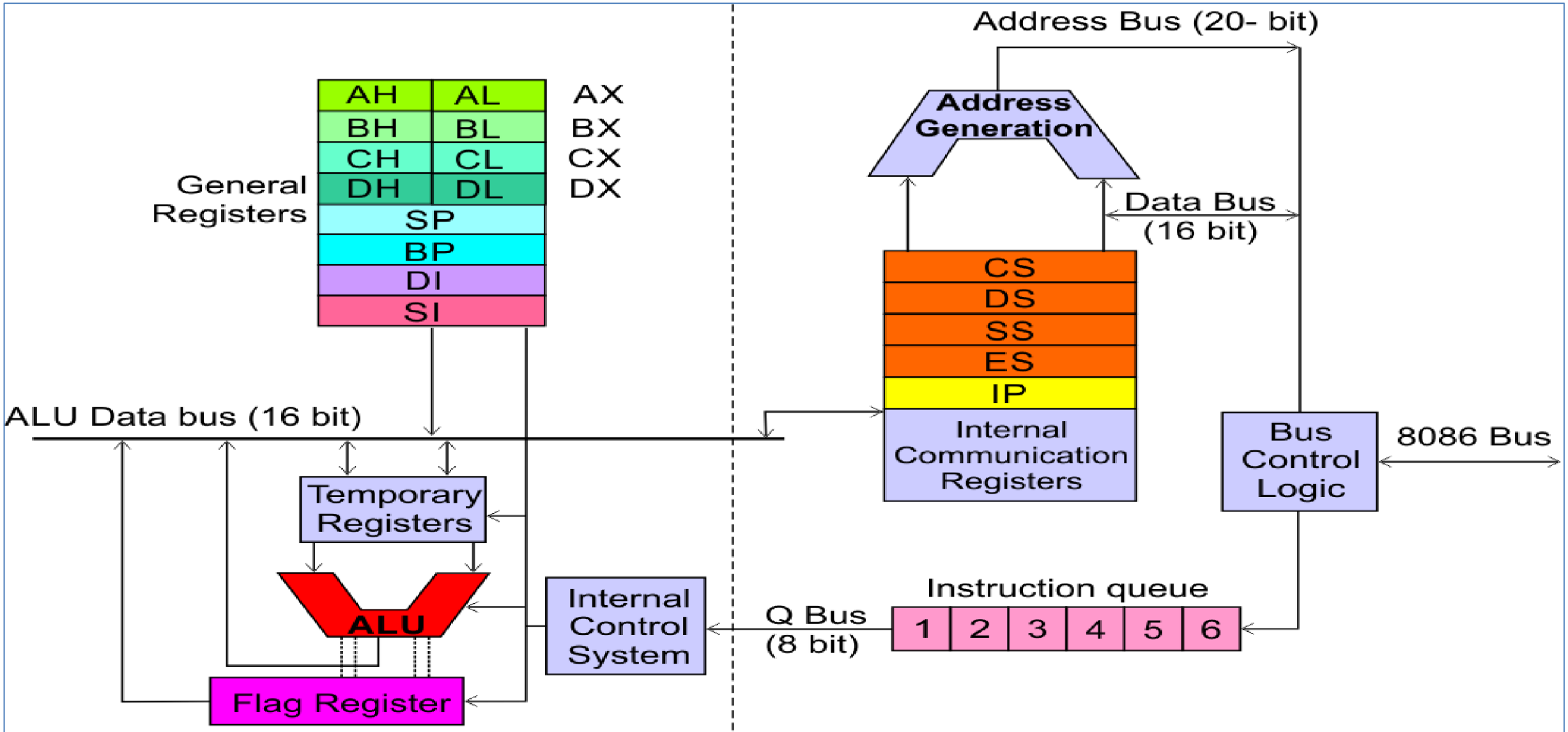
- Thus, Intel split the internal structure of 8086 into two sections:
  - **Execution Unit (EU)**

    Executes instructions already fetched
  - **Bus Interface Unit (BIU)**

    Accesses memory and peripherals



# 8086 Hardware Architecture




**Execution Unit (EU)**

EU executes instructions that already fetched by BIU. BIU and EU functions separately.

**Bus Interface Unit (BIU)**

Reads (fetch) instructions, reads operands and writes results.

# Bus Interface Unit (BIU)

- BIU Principle Functionality
  - BIU fetches instructions, reads data from memory and I/O ports, writes data to memory and I/ O ports.
  - Then, EU executes instructions that already fetched by BIU
- BIU Components
  - FIFO Queue to Prefetch up to 6 instruction bytes from external memory.
  - Dedicated adder for address translations.
  - Bus Control Logic unit to generate all bus control signals.
  - IP & Four 16-bit segment registers: CS, DS, SS, ES.
- Instruction queue


  - FIFO Queue pre-fetch up to 6 bytes of instruction from the memory ahead of time.
  - Speed up the execution➔ Overlapping instruction fetch with execution (**Pipelining**).

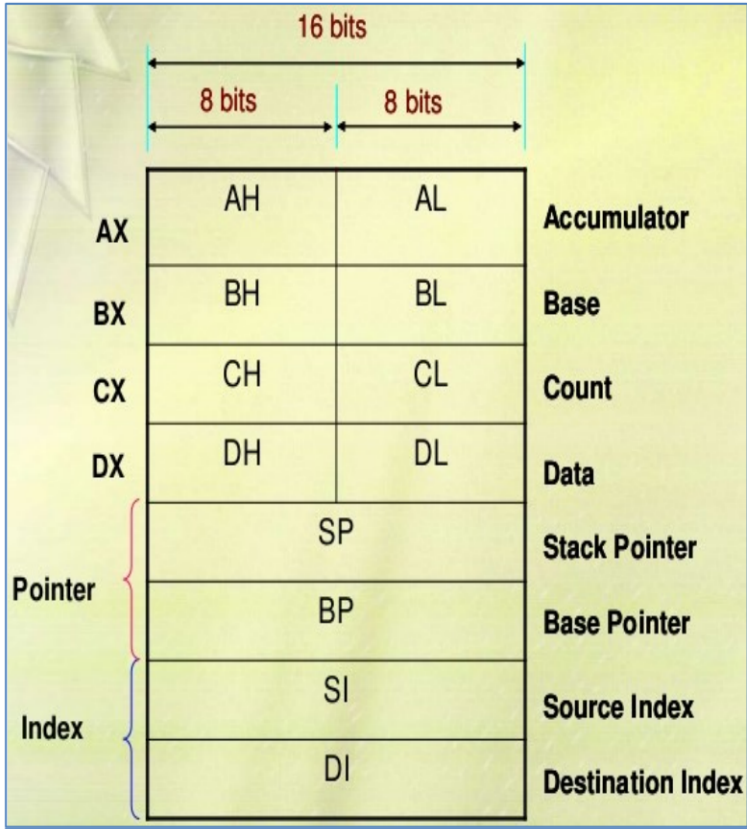
# Bus Interface Unit (BIU)

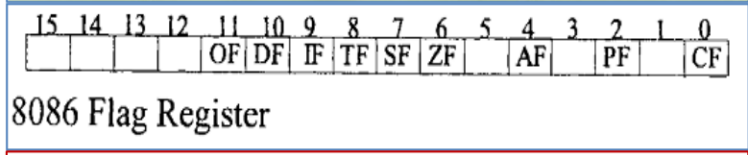
- BIU Registers
  - IP: 16-bit Instruction Pointer (offset points to current instruction).
  - CS: 16-bit Code Segment Register (points to current code segment).
  - DS: 16-bit Data Segment Register (points to current data segment).
  - SS: 16-bit Stack Segment Register (points to current stack segment).
  - ES: 16-bit Extra Segment Register (points to current extra segment).

- How BIU do Physical Address Calculation!
  - This differ according to the addressing methods, for instance:
    - For memory data: address calculated from [DS: BX] or [DS: SI].
    - For memory instructions: address calculated from [CS: IP].
    - For Stack instructions: address calculated from [SS: SP].
    - For String instructions: address calculated from [ES: DI].

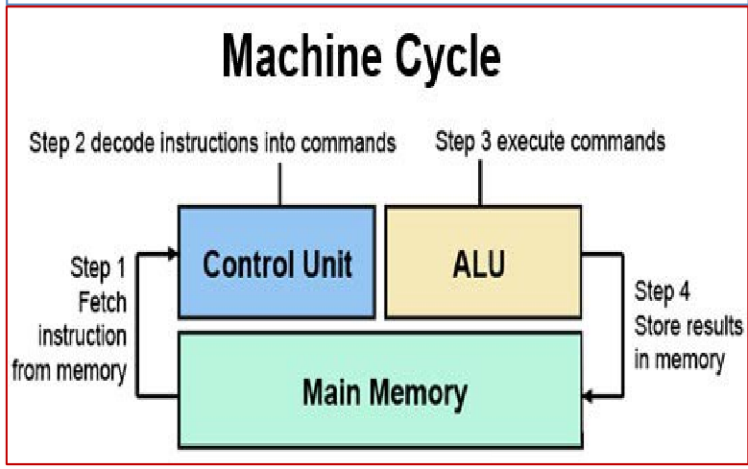
# Execution Unit

- EU Principle Functionality
  - Decodes & Executes instructions already fetched by BIU.
  - Its also responsible of generation the control signals of MP.
- EU Components
  - Control Unit to generate the control signals.
  - 16-bit ALU for arithmetic and logic operations.
  - Four 16-bit general purpose registers: AX, BX, CX, DX.
  - Two 16-bit index registers (SI, DI)
  - Two 16-bit pointer registers (SP, BP).
  - One 16-bit Flags register (9 active, 7 are reserved).
- Control Unit (CU).
  - Read & Decode instructions from program Memory.
  - Generate control signals via control bus.




8086 Flag Register



# Execution Unit

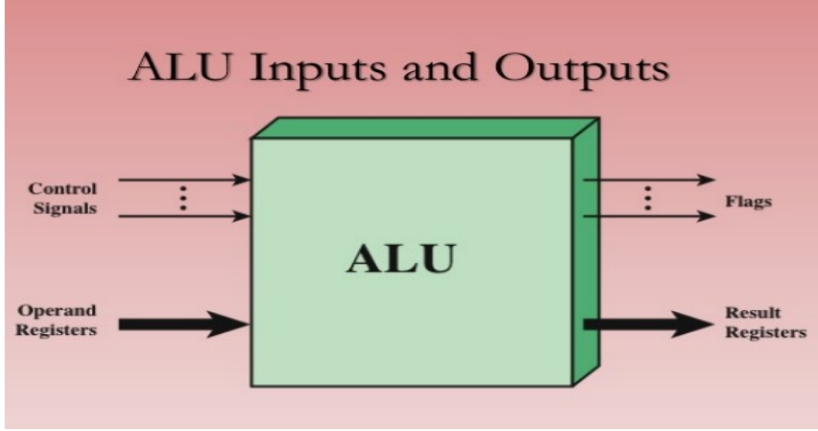
- General Purpose Registers (GPRs)
  - AX (Accumulator): used IN/OUT instructions, MUL and DIV instructions.
  - BX (Base): used for memory addressing and operands.
  - CX (Counter): used mainly by SHIFT, ROTATE, and LOOP instructions.
  - DX (Data): used mainly to hold one of the following:
    - ▶High 16-bit result after 16x16 bit MUL (LOW 16-bit in AX)
    - ▶High 16-bit dividend before a 32÷16 DIV (LOW 16-bit in AX)
    - ▶16-bit remainder after 32÷16 DIV (16-bit quotient in AX).
- Pointer Registers (SP/BP).
  - Stack Pointer & Base Pointer are used to access data in the stack segment.
  - SP is to used as an offset access STACK memory with SS as segment register.
  - SP is auto- incremented or decremented due to execution stack instructions.
  - BP is used by the user in the based addressing mode (later).

# Execution Unit

- Index Registers (SI and DI).
  - Source index and destination index are used with string Instructions along with DS & ES, respectively.
- Flags Register (FL)
  - FL bits are set or reset by EU to reflect the results of ALU.
  - DF: Controlling string operations.
  - IF: Controlling Maskable interrupts.
  - TF : Provides Single-Step debugging.


| | | | |
|---|---|---|---|
| 1. | CF | CARRY FLAG | **Conditional Flags (Compatible with 8085, except OF)** These Flags are also called status bits |
| 2. | PF | PARITY FLAG | |
| 3. | AF | AUXILIARY CARRY | |
| 4. | ZF | ZERO FLAG | |
| 5. | SF | SIGN FLAG | |
| 6. | OF | OVERFLOW FLAG | |
| 7. | TF | TRAP FLAG | **Control Flags** Those can be set or cleared By Programmer |
| 8. | IF | INTERRUPT FLAG | |
| 9. | DF | DIRECTION FLAG | |

# Execution Unit




- Arithmetic Logic Unit (ALU).
  - ALU contains circuitry to perform arithmetic and logic operations.
  - ALU controls individual bits of Flags register (set (1) or clear (0)).
  - This happens as a response once ALU completes a specific operation.
  - For example: if the result was zero or non-zero, if it was positive or negative, or if the result is too big to be stored in a byte or word.
  - EU checks the status Flag bits when executing conditional jumps.
  - Size of ALU conforms to word length of MP (16-bit MP have 16-bit ALU.

# Thank you