8086 Features

- 16-bit Arithmetic Logic Unit
- 16-bit data bus (8088 has 8-bit data bus)
- 20-bit address bus - 2^{20} = 1,048,576 = 1 meg

The address refers to a byte in memory. In the 8088, these bytes come in on the 8-bit data bus. In the 8086, bytes at even addresses come in on the low half of the data bus (bits 0-7) and bytes at odd addresses come in on the upper half of the data bus (bits 8-15).

The 8086 can read a 16-bit word at an even address in one operation and at an odd address in two operations. The 8088 needs two operations in either case.

The least significant byte of a word on an 8086 family microprocessor is at the lower address.

8086 Architecture

- **The 8086 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.**

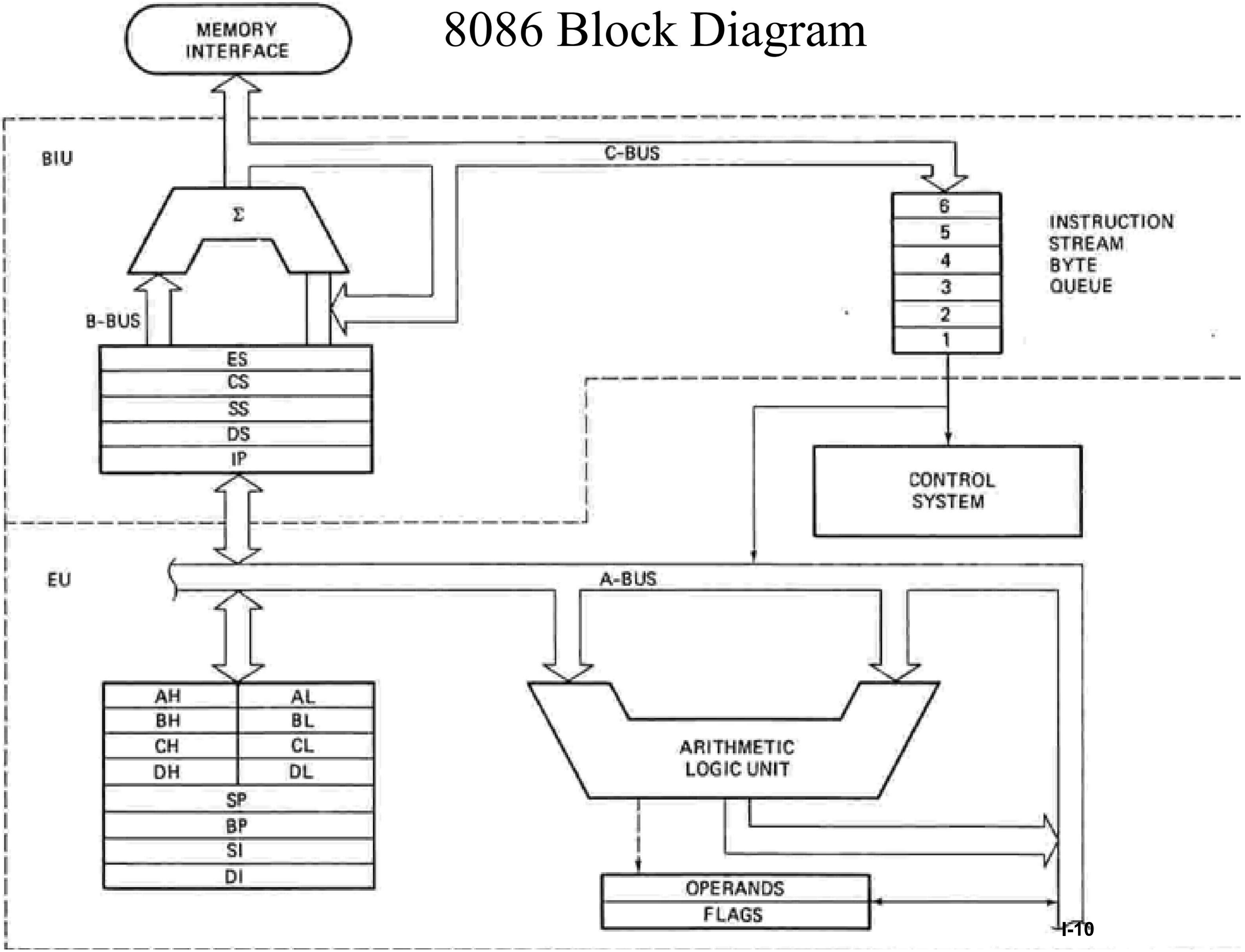
- **The EU decodes and executes the instructions using the 16-bit ALU.**

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

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

The BIU fetches instructions using the CS and IP, written CS:IP, to contruct 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.

8086 Block Diagram



8086 Architecture

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

Flag Register

- Flag register contains information reflecting the current status of a microprocessor. It also contains information which controls the operation of the microprocessor.

15														0
—	NT	IOPL	OF	DF	IF	TF	SF	ZF	—	AF	—	PF	—	CF

- Control Flags

IF:	Interrupt enable flag
DF:	Direction flag
TF:	Trap flag

- Status Flags

CF:	Carry flag
PF:	Parity flag
AF:	Auxiliary carry flag
ZF:	Zero flag
SF:	Sign flag
OF:	Overflow flag
NT:	Nested task flag
IOPL:	Input/output privilege level

Flags Commonly Tested During the Execution of Instructions

- There are five flag bits that are commonly tested during the execution of instructions
  - Sign Flag (Bit 7), SF: 0 for positive number and 1 for negative number
  - Zero Flag (Bit 6), ZF: If the ALU output is 0, this bit is set (1); otherwise, it is 0
  - Carry Flag (Bit 0), CF: It contains the carry generated during the execution
  - Auxiliary Carry, AF: (Bit 4) Depending on the width of ALU inputs, this flag bit contains the carry generated at bit 3 (or, 7, 15) of the 8088 ALU
  - Parity Flag (bit2), PF: It is set (1) if the output of the ALU has even number of ones; otherwise it is zero

Direction Flag

- Direction Flag (DF) is used to control the way SI and DI are adjusted during the execution of a string instruction

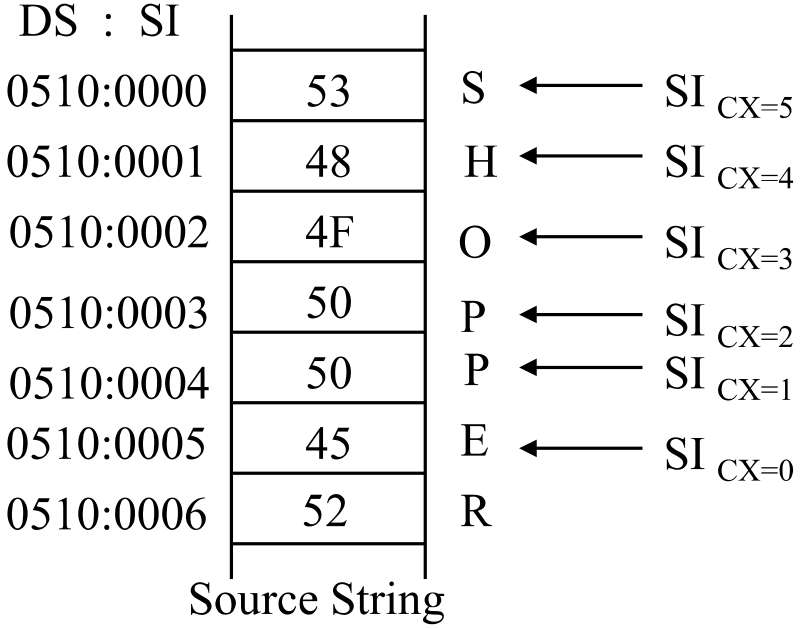
  — DF=0, SI and DI will auto-increment during the execution; otherwise, SI and DI auto-decrement

  — Instruction to set DF: **STD**; Instruction to clear DF: **CLD**

  — Example:

```
CLD
MOV CX, 5
REP MOVSB
```

At the beginning of execution,
DS=0510H and SI=0000H



DS : SI			
0510:0000	53	S	SI $_{CX=5}$
0510:0001	48	H	SI $_{CX=4}$
0510:0002	4F	O	SI $_{CX=3}$
0510:0003	50	P	SI $_{CX=2}$
0510:0004	50	P	SI $_{CX=1}$
0510:0005	45	E	SI $_{CX=0}$
0510:0006	52	R	

Source String

8086 Programmer’s Model

BIU registers
(20 bit adder)

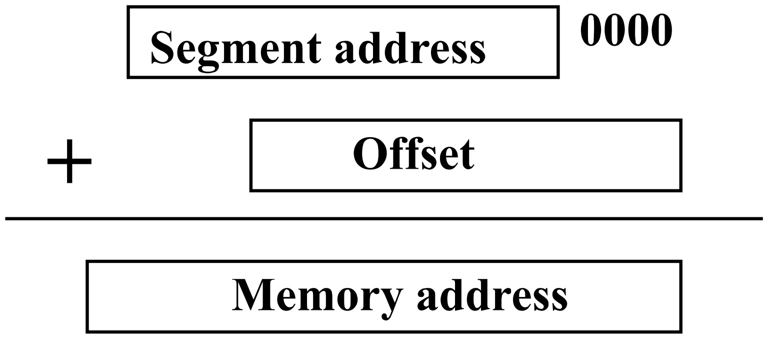
ES	Extra Segment
CS	Code Segment
SS	Stack Segment
DS	Data Segment
IP	Instruction Pointer

EU registers
16 bit arithmetic

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		

Memory Address Calculation

- ❑ Segment addresses must be stored in segment registers
- ❑ Offset is derived from the combination of pointer registers, the Instruction Pointer (IP), and immediate values



Segment address 0000

+ Offset

Memory address

- ❑ Examples

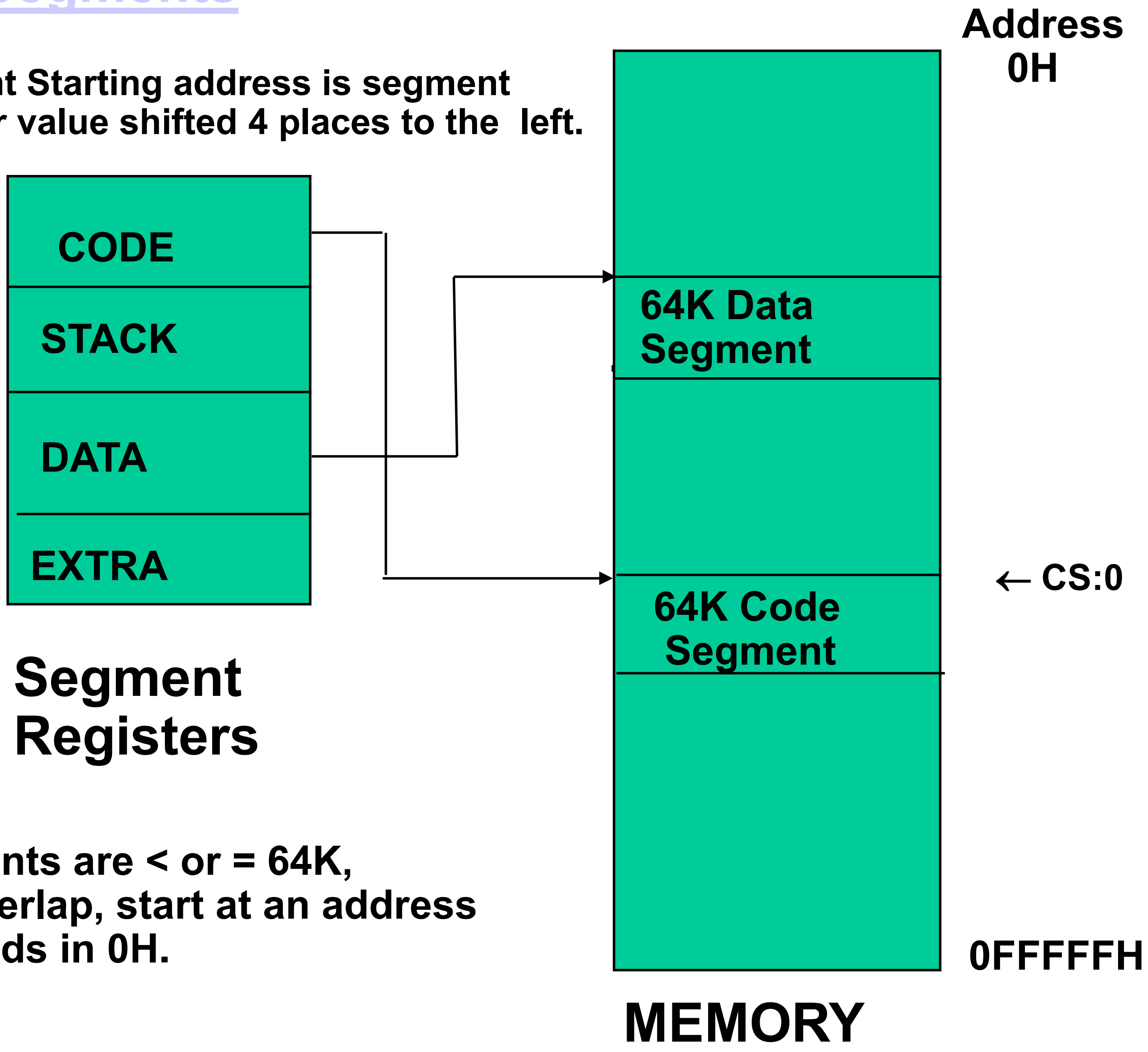
CS	3	4	8	A	0
IP +		4	2	1	4
Instruction address	3	8	A	B	4

SS	5	0	0	0	0
SP +		F	F	E	0
Stack address	5	F	F	E	0

DS	1	2	3	4	0
DI +		0	0	2	2
Data address	1	2	3	6	2

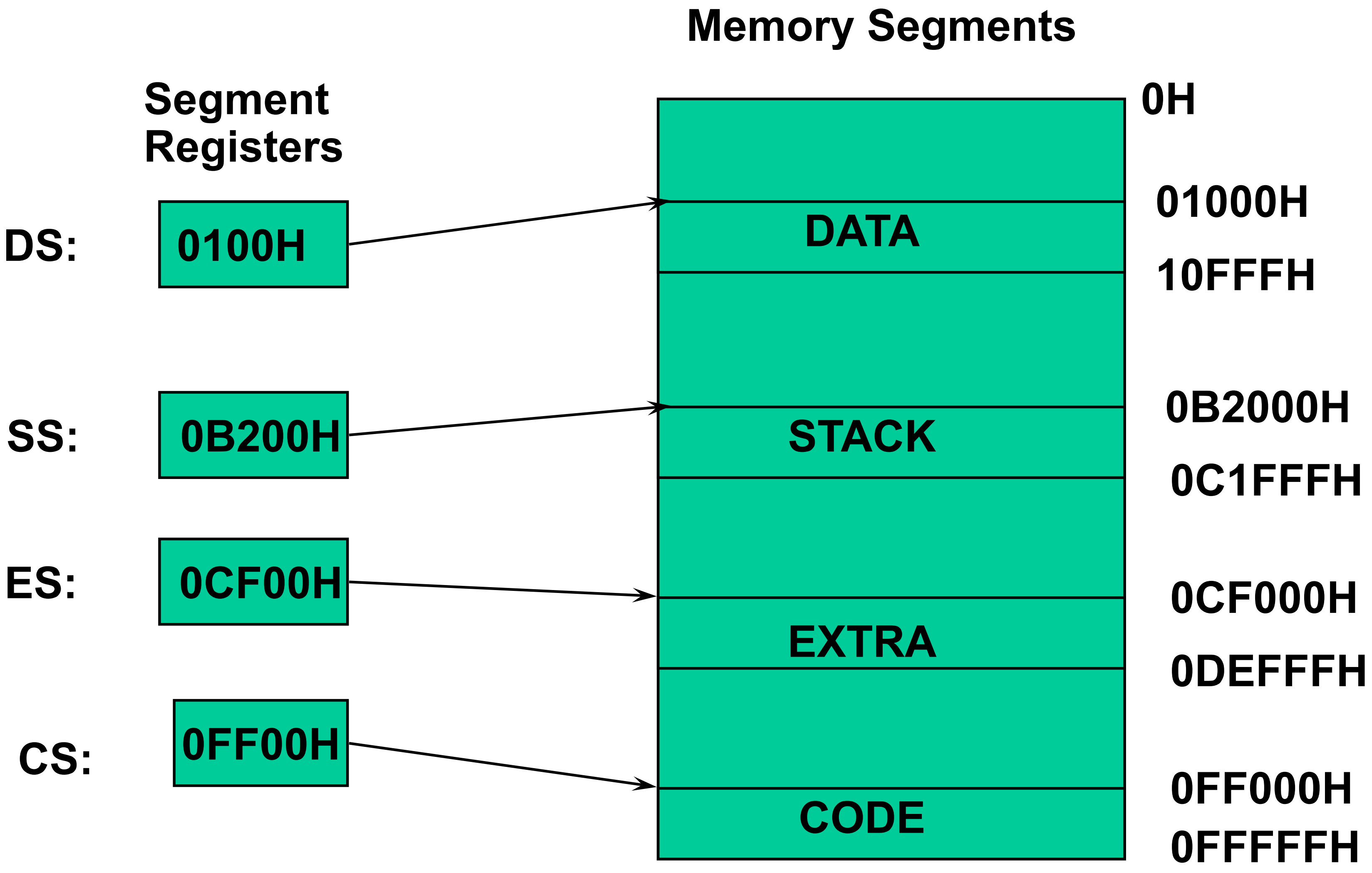
Segments

Segment Starting address is segment register value shifted 4 places to the left.



Segments are < or = 64K,
can overlap, start at an address
that ends in 0H.

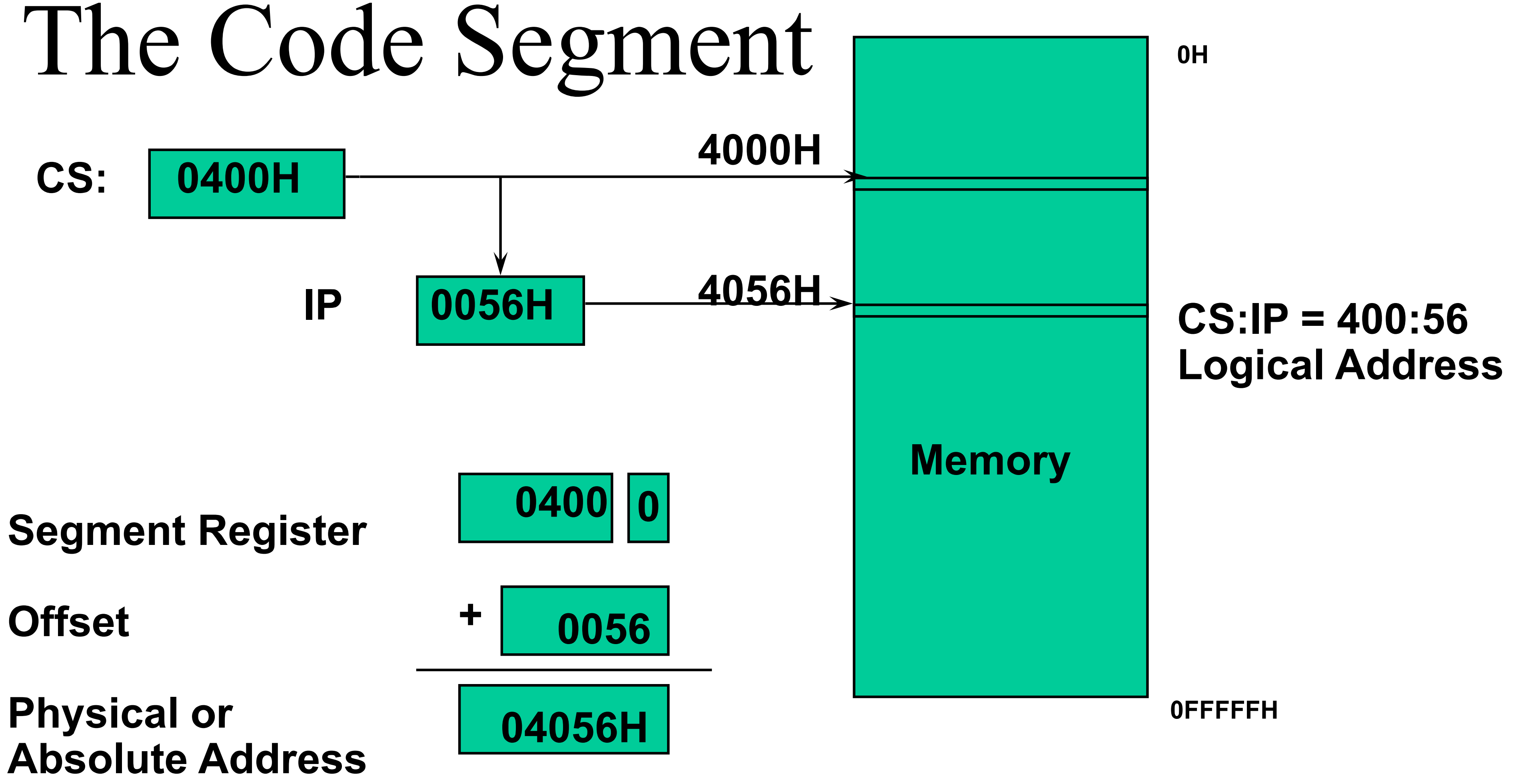
8086 Memory Terminology



Segments are < or = 64K and can overlap.

Note that the Code segment is < 64K since 0FFFFFH is the highest address.

The Code Segment



CS: 0400H

4000H

IP 0056H

4056H

0H

CS:IP = 400:56
Logical Address

Memory

0FFFFFH

Segment Register 0400 0

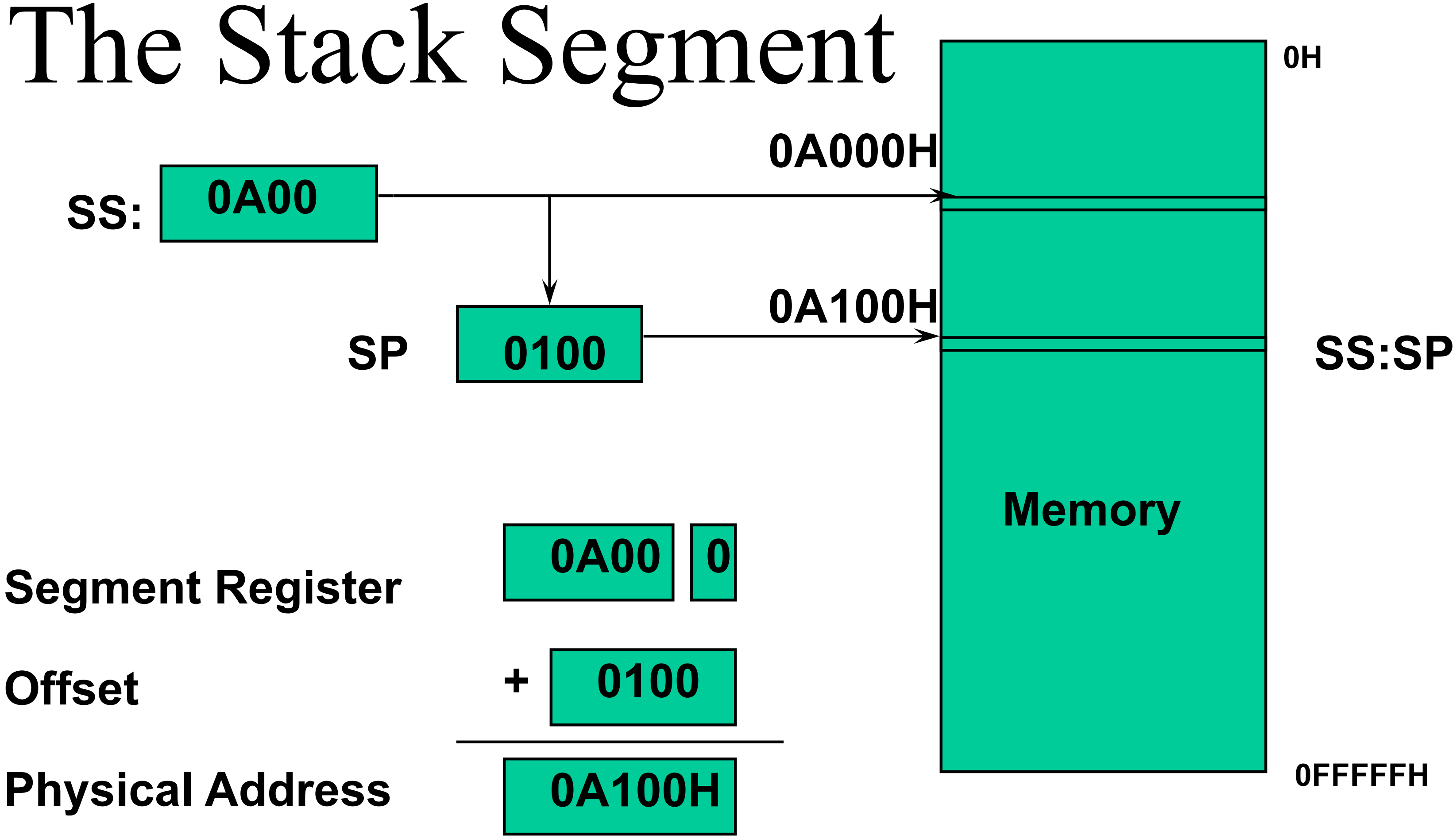
Offset + 0056

Physical or
Absolute Address 04056H

The offset is the distance in bytes from the start of the segment.
The offset is given by the IP for the Code Segment.
Instructions are always fetched with using the CS register.

The physical address is also called the absolute address.

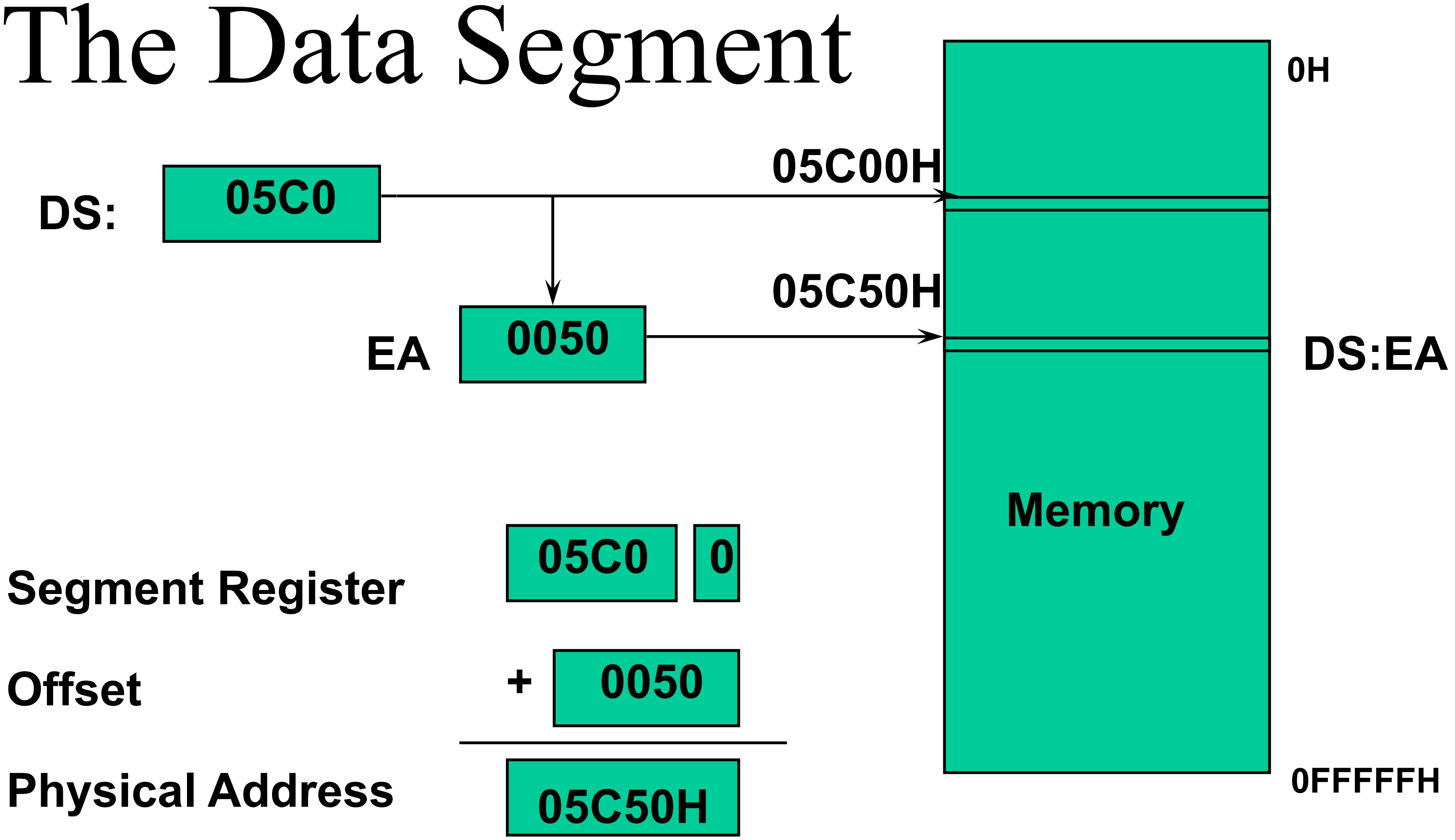
The Stack Segment



The offset is given by the SP register.
The stack is always referenced with respect to the stack segment register.
The stack grows toward decreasing memory locations.
The SP points to the last or top item on the stack.

PUSH - pre-decrement the SP
POP - post-increment the SP

The Data Segment



Data is usually fetched with respect to the DS register.
The effective address (EA) is the offset.
The EA depends on the addressing mode.