Chapter Three

8086 Addressing Modes

Compiled by Yilkal B.

Chapter Outline

- *Addressing modes:*
  - *Definition and classification.*
- *Data addressing modes.*
- *Program memory addressing modes*
- *Stack memory addressing modes.*

Addressing Modes: Definition and classification

Addressing Modes:

- Addressing mode provide different ways for access an address to **given data** to a processor.
- **When 8086 executes an instruction**, it performs the specified function on data.
- **Operated data** is stored in the memory location.
- There are **various techniques to specify address of data**. These techniques are **called Addressing Modes.**

Classification of Addressing Modes:

I. Data-Addressing Modes:

- This mode is related to **data transfer operation**, that is, data is transferred either from the **memory** to **internal registers** of 8086 processors or from **one register** to **another register**.

  Example: MOV AX, DX

Cntd…

II. Program Memory addressing Modes:

- This mode involves **program memory addresses during various operations**.
- Example: JMP AX, in this instruction, the **code execution** control jumps to **the current code segment location addressed** by the contents of AX register.

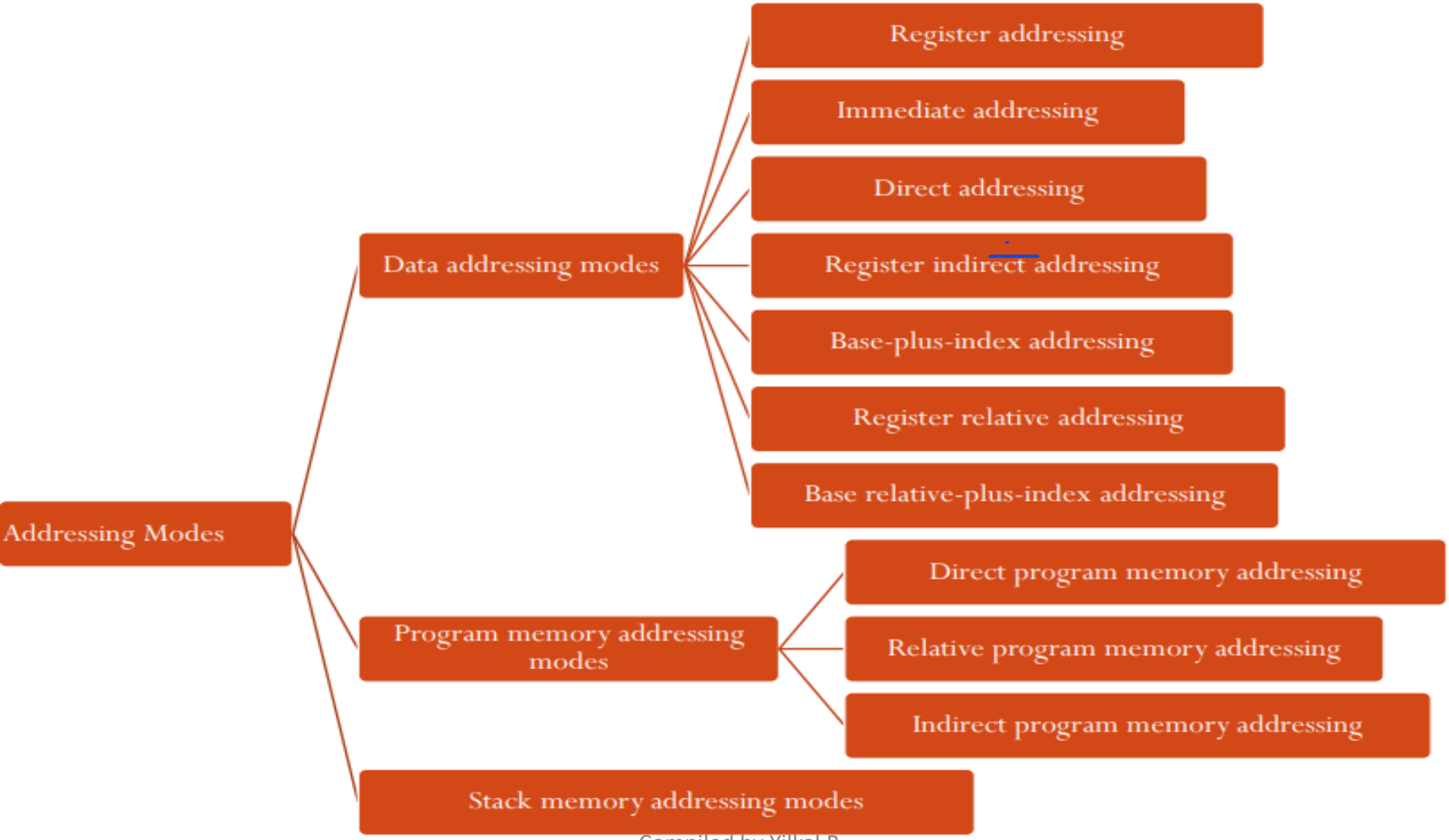
III. Stack memory addressing Modes:

- This mode involves **stack registry operations**.
- Example:

1. PUSH AX, this instruction copies the contents of AX register to the stack
2. POP AX, this instruction move the contents of AX register from the stack

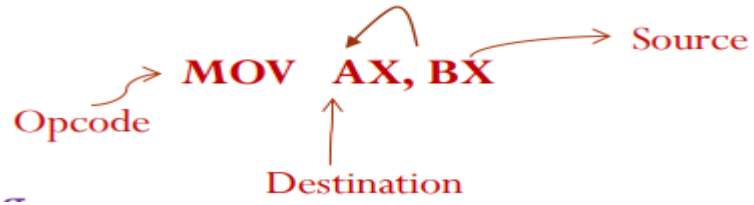
Cntd…

Addressing Modes: classification



I. Data addressing modes

❖ *Note: We will use MOV instruction to explain all the data addressing modes.*



1. Register addressing

➢ *Register addressing transfers a **copy of a byte or word** from the **source register** to **destination register**.*

➢ *Register addressing is the **most common form of data addressing**.*

➢ *8-bit register names with register addressing: AH, AL, BH, BL, CH, CL, DH, DL.*

➢ *16-bit register names: AX, BX, CX, DX, SP, BP, SI ,DI, IP, CS, SS, DS and ES.*

➢ ***Never mix an 8-bit register with 16-bit**, it **is not allowed** in microprocessor.*

➢ *Code segment register (CS) is never used as destination.*

Cntd…

❖ ***Segment*** *to* ***segment*** *MOV instruction* ***is not allowed.***

E.g.,

1. ***MOV AL, BL*** *; Copies 8-bit content of BL into AL*
2. ***MOV AX, CX*** *; Copies 16-bit content of CX into AX*
3. ***MOV ES, DS*** *; Not allowed (segment to segment)*
4. ***MOV BL, DX*** *; Not allowed (mixed size)*
5. ***MOV CS, AX ;*** *Not allowed (Code segment register may not be destination register)*

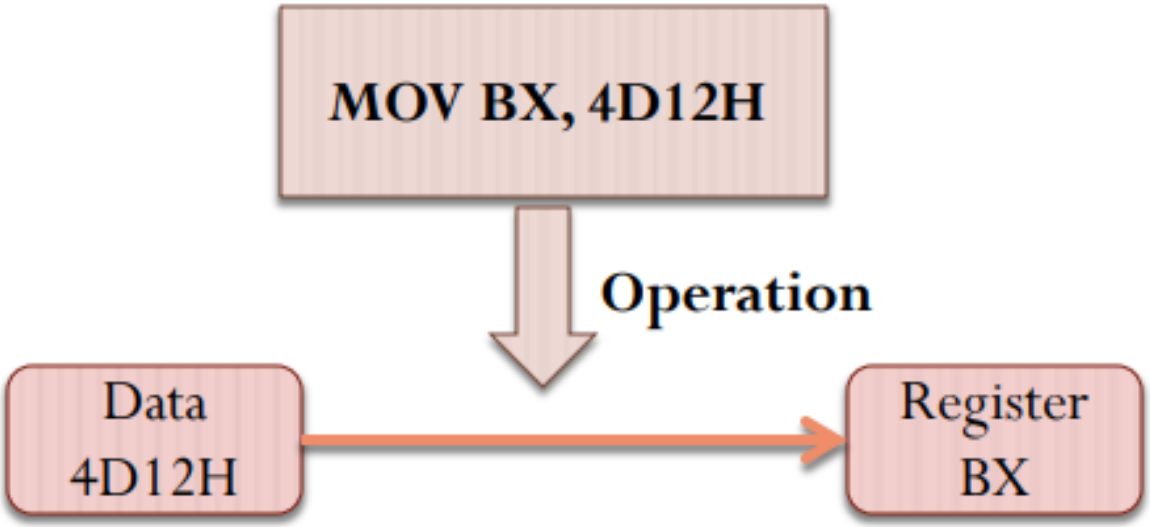
2. Immediate addressing: *It* ***transfers the source****, an* ***immediate byte or word data****, into the destination register.*

- ***Immediate data*** *means constant data, whereas data transferred from a register or memory location are* ***variable data.***

Cntd…

E.g.,

1. ***MOV BL, 44*** *; Copies 44 decimal (2CH) into BL*
2. ***MOV AX, 44H*** *; Copies 0044H into AX*
3. ***MOV AL, 'A'*** *; Copies ASCII A into AL*



Cntd…

3. Direct data addressing: *It moves a* ***byte or word between*** *a data segment memory location and* ***register****.*

- *The instruction set* ***does not*** *support a* ***memory to memory*** *transfer except with the MOV instruction.*
- *Direct addressing with a MOV instruction* ***transfers data between*** *a memory location, located within the data segment, and the AL (8-bit) or, AX (16-bit).*
- *A MOV instruction using this type of addressing is usually* ***a three byte long instruction.***
- *The effective address is given directly in the source.*

E.g., MOV AL, DS:[1234H]

Cntd…

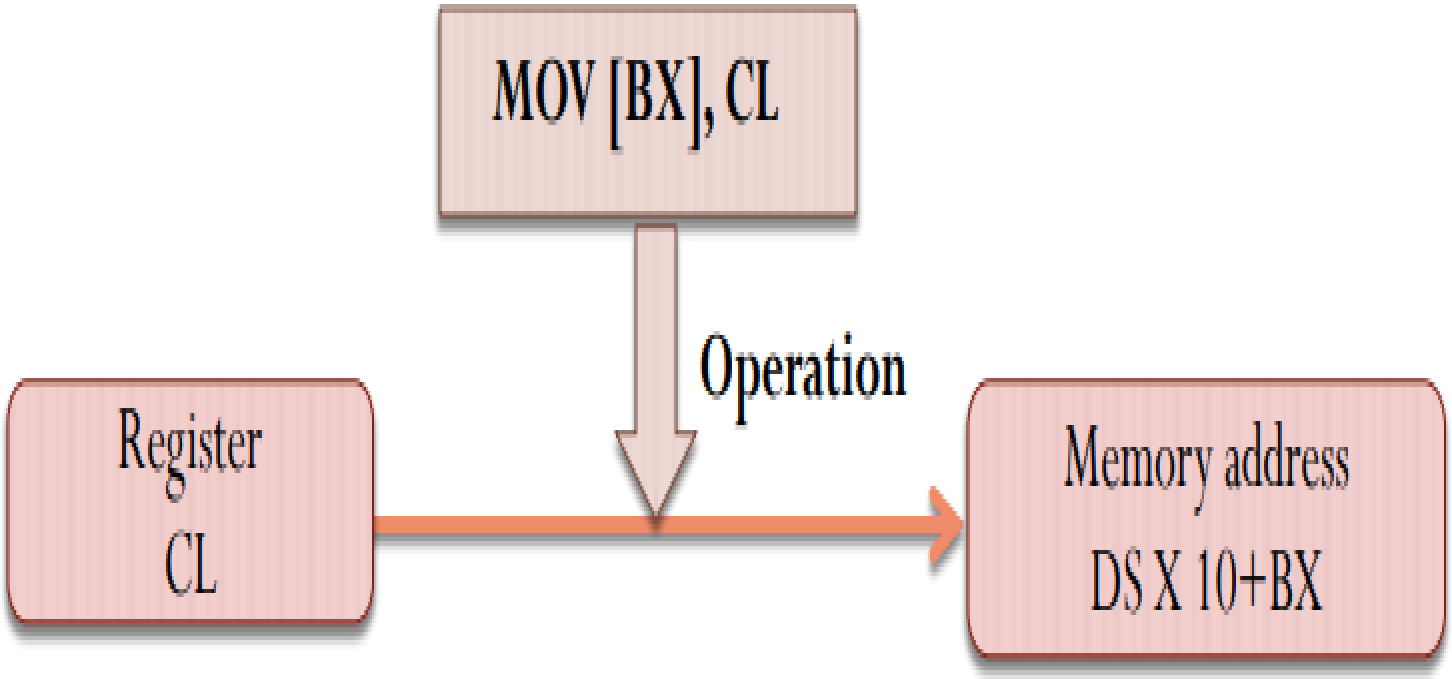
4. Register Indirect addressing

- Register addressing transfers a **byte or word** between a register and memory location addressed by **an index** or **base register**.
- The index and base registers are BP, BX, DI and SI. these registers hold the offset address of the memory location.
- The **data segment is used** by default with register **indirect addressing** or any other addressing modes that uses BX, DI or, SI to address memory.
- If BP register addresses memory, the stack segment is used by default.
- The [ ] symbol denote **indirect addressing** in assembly language.

Cntd…

E.g.,

1. *MOV CX, [BX] ; Copies the word contents of the **data segment memory** location addressed by BX into CX.*



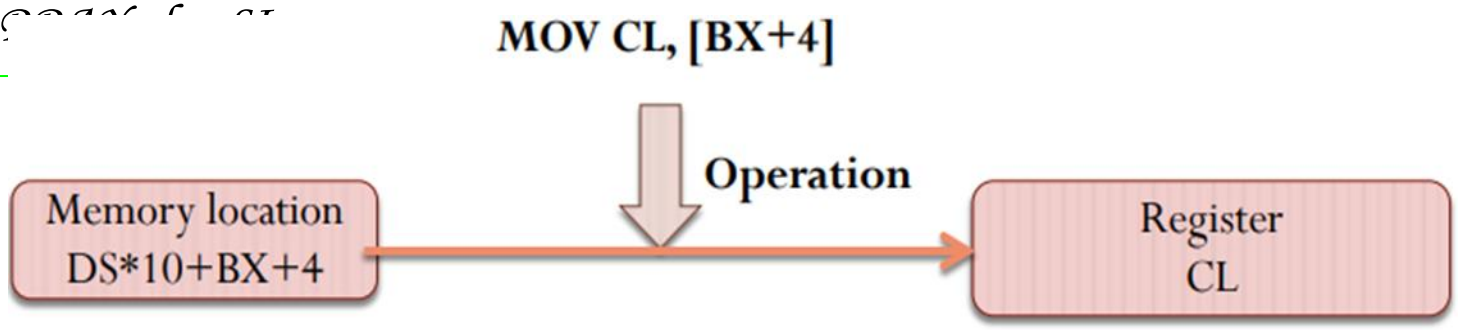
Cntd…

5. Register Relative addressing/ Base Address mode : *It moves a byte or word between a* ***register*** *and the* ***memory location*** *addressed by an index or base register (BP, BX, DI or SI) plus a displacement.*

- *Remember that BX, DI or SI addresses* ***data segment*** *and BP addresses the stack segment.*

E.g.,

1. *MOV AX, [DI+100H] ; Copies the word content of the data segment memory location addressed by DI plus 100H into AX.*
2. *MOV ARRAY[SI], BL ; Copies BL into the data segment memory location addressed by*



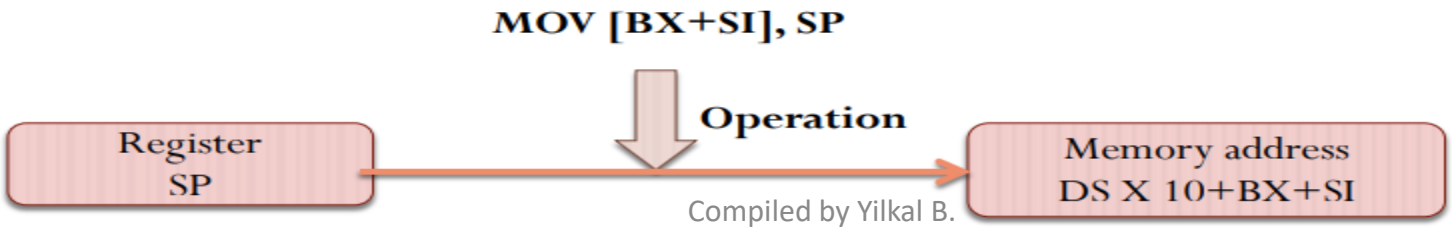
Cntd…

6. Base-Plus-Index addressing: *it transfer a byte or word between a* ***register*** *and a* ***memory location*** *addressed by a* ***base register*** *(BP or BX)* ***plus*** ***an index register*** *(DI or SI).*

- *The base register often* ***holds the beginning location of a memory array,***
- *Whereas, the index register holds the* ***relative position of*** *an element in the array.*
- *When BP addresses memory, stack segment is selected by default and when* ***BX*** *addresses memory data segment is selected by default.*

E.g.,

1. ***MOV CX, [BX+DI];*** *Copies the word content of the data segment memory location addressed by BX plus DI into CX.*
2. ***MOV [BP+DI], AH ;*** *Copies AH into stack segment memory location addressed by BP plus DI*



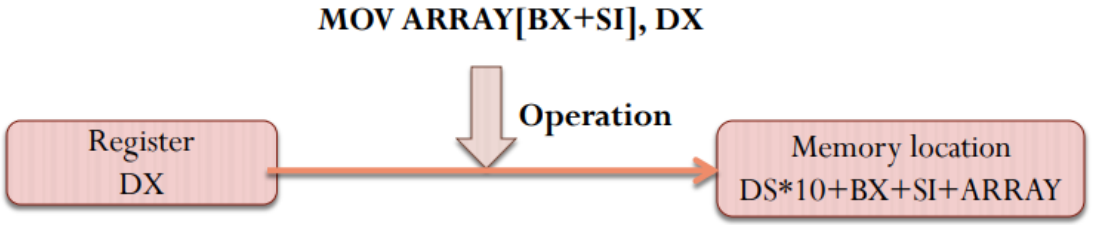
Cntd…

7. Base Relative plus Index addressing : *it transfers* ***a byte or word*** *between a register and a memory location addressed by a base and an index register plus displacement.*

- ***It is similar to base plus index addressing****, but it* ***adds a displacement*** *beside using a base register and an index register.*
- *This type of addressing mode often addresses a* ***two-dimensional array of memory*** *data.*

E.g., *1.* ***MOV DH, [BX+DI+20H]*** *; Copies the byte content of data segment memory location addressed by the sum of BX, DI and 20H into DH.*

2. ***MOV LIST[BP+DI],*** *CL ; Copies CL into the stack segment memory location addressed by the sum of LIST, BP, SI and*



- *Scaled-Index Addressing and RIP Relative Addressing(reading Assignment)*

II. Program Memory addressing Modes

- *The Program Memory Addressing mode* ***is used in branch instructions.***
- *These branch instructions are instructions which are* ***responsible for changing the regular flow of the instruction execution*** *and* ***shifting the control to some other location.***
- *In 8086 microprocessor, these instructions are usually* ***JMP*** *and* ***CALL*** *instructions.*
- *The Program memory Addressing Mode contains further* ***three addressing modes*** *within it. They are:*

  a. *Direct Program memory Addressing*

  b. *Indirect Program memory Addressing*

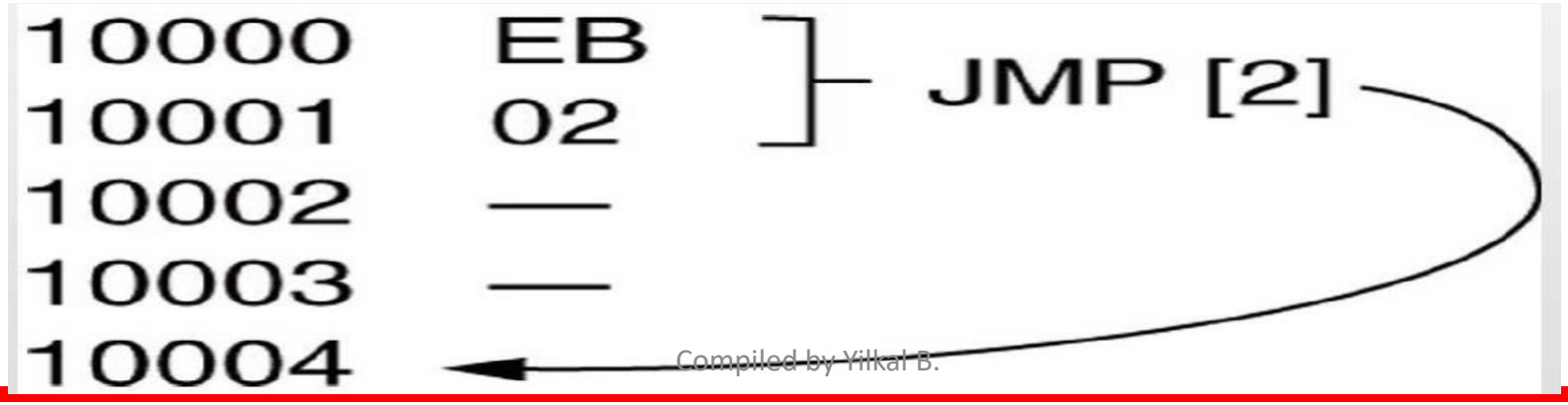
  c. *Relative Program memory Addressing*

a. Direct Program Memory Addressing

- *Used for all jumps and calls by **early microprocessor**; also used in high-level languages, such as BASIC.*
- *The microprocessor uses this form, but **not as often as** relative and indirect program memory addressing.*
- *The instructions for **direct program memory addressing** store the address with the opcode.*
- *Often called **a far jump** because it can jump to any memory location for the next instruction.*
- *The only other instruction using direct program addressing is the intersegment or **far CALL** instruction.*
- *Usually, **the name of a memory address**, called **a label,** refers to the location that is called or jumped to instead of the actual numeric address.*

6. Relative Program Memory Addressing

- ***Not available in all early microprocessors***, *but it is available to this family of microprocessors.*
- *The term* ***relative*** *means "relative to the instruction pointer (IP)".*
- *The JMP instruction is a 1-byte instruction, with a 1-byte or a 2-byte displacement that adds to the instruction pointer.*
- *A JMP [2] instruction. This instruction skips over the 2 bytes of memory that follow the JMP instruction.*



c. Indirect Program Memory Addressing

- The microprocessor allows several forms of program **indirect memory addressing** for the JMP and CALL instructions.
- In 80386 and above, an extended register can be used to hold the address or indirect address of a relative JMP or CALL.
- If a **relative register** holds **the address**, the jump is considered to be an **indirect jump**.
- For example, JMP [BX] refers to the memory location within the data segment at the offset address contained in BX.
  - at this offset address is a 16-bit number used as the offset address in the intersegment jump
  - this type of jump is sometimes called an **indirect indirect** or **double-indirect jump**

III. Stack Memory-Addressing Modes

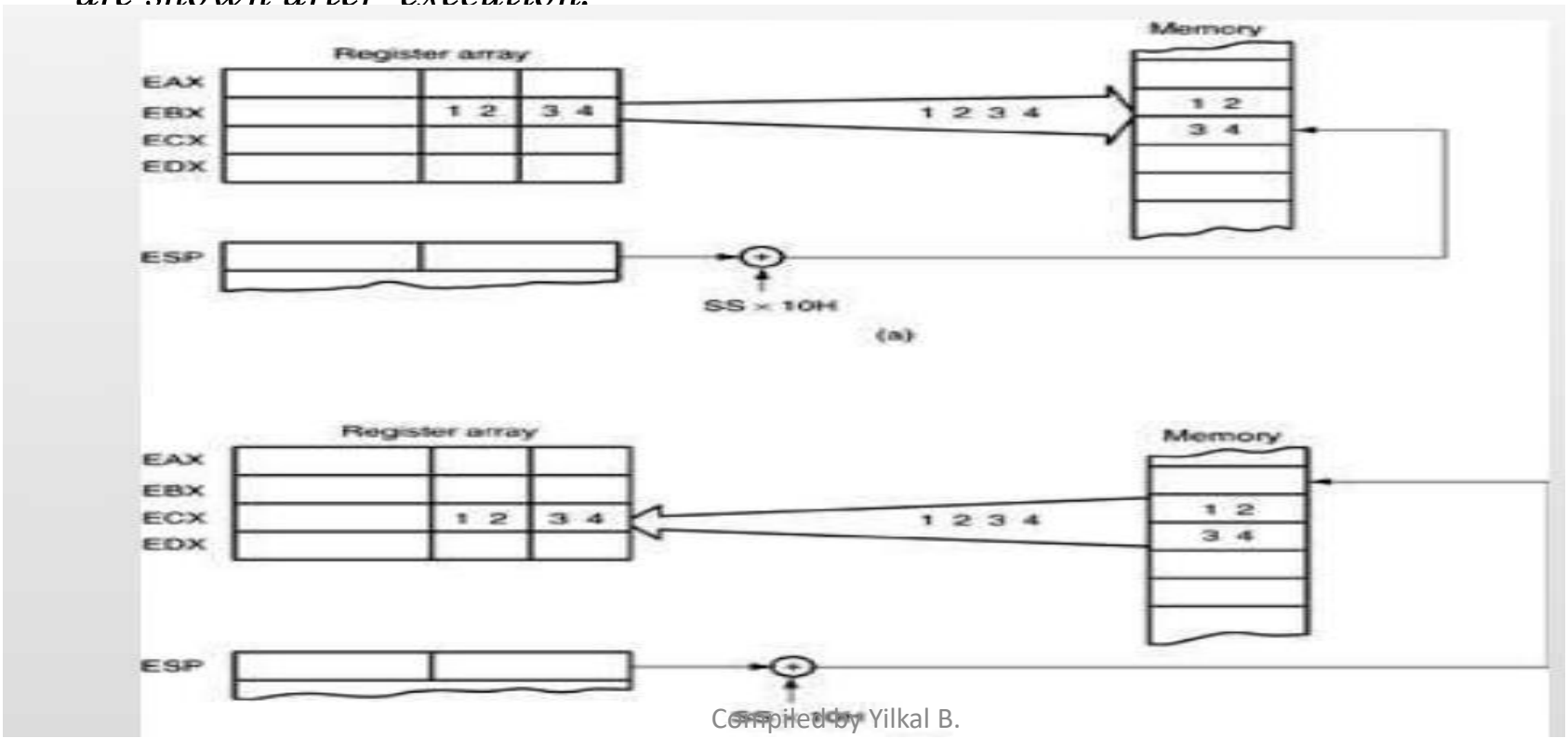
- The stack plays an important role in all microprocessors.
  - **holds data temporarily and stores return addresses used by procedures**
- Stack memory is LIFO (last-in, first-out) memory
  - **describes the way data are stored and removed from the stack**
- Data are **placed on the stack** with a **PUSH** instruction; **removed** with **a POP** instruction.
- Stack memory is maintained by **two registers**:
  - the stack pointer (SP or ESP)
  - the stack segment register (SS)
- **Whenever a word of data is pushed onto the stack,:**
  - The high-order 8 **bits** are placed in the location addressed by **SP – 1**.
  - The **low-order** 8 bits are placed in the location addressed by **SP – 2**

Cntd…

E.g., The PUSH and POP instructions:

(a) PUSH BX places the contents of BX onto the stack;

(b) POP CX removes data from the stack and places them into CX. Both instructions are shown after execution.



End of Chapter Three

Any 3.1.8 Scaled-Index Addressing

3.1.9 RIP Relative Addressing *Question?*