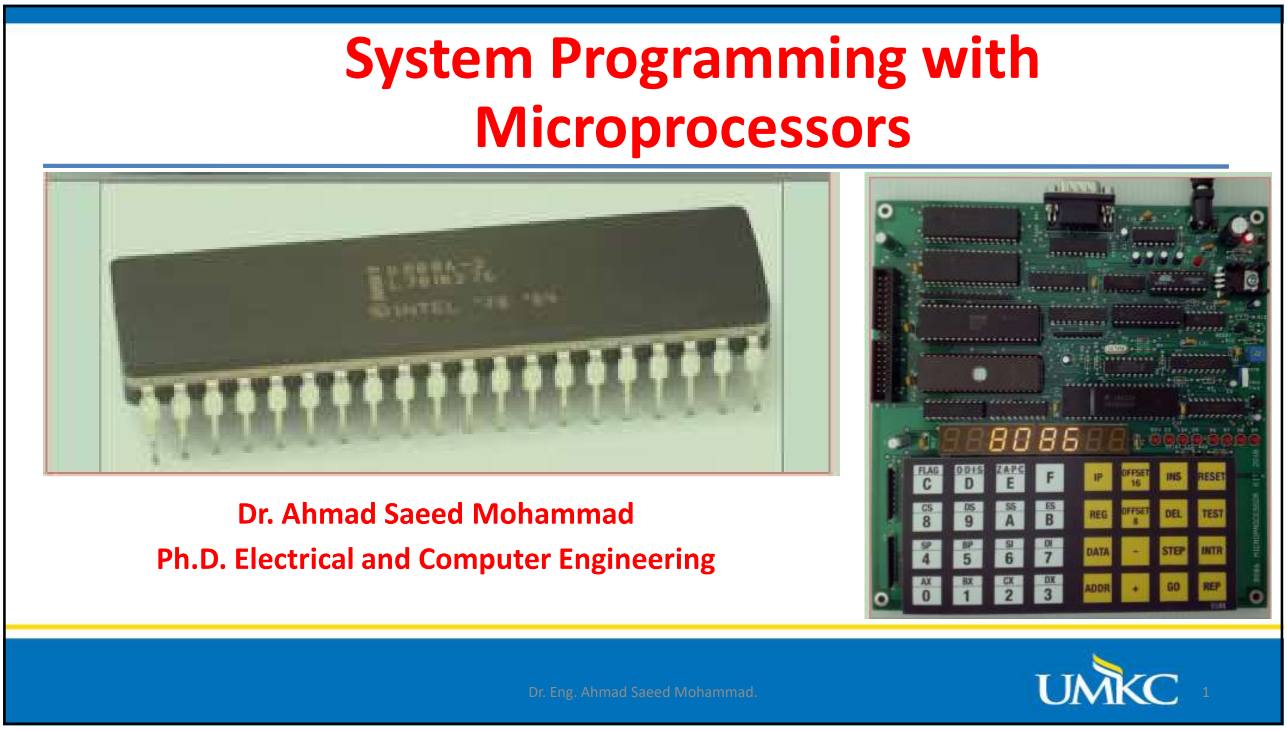


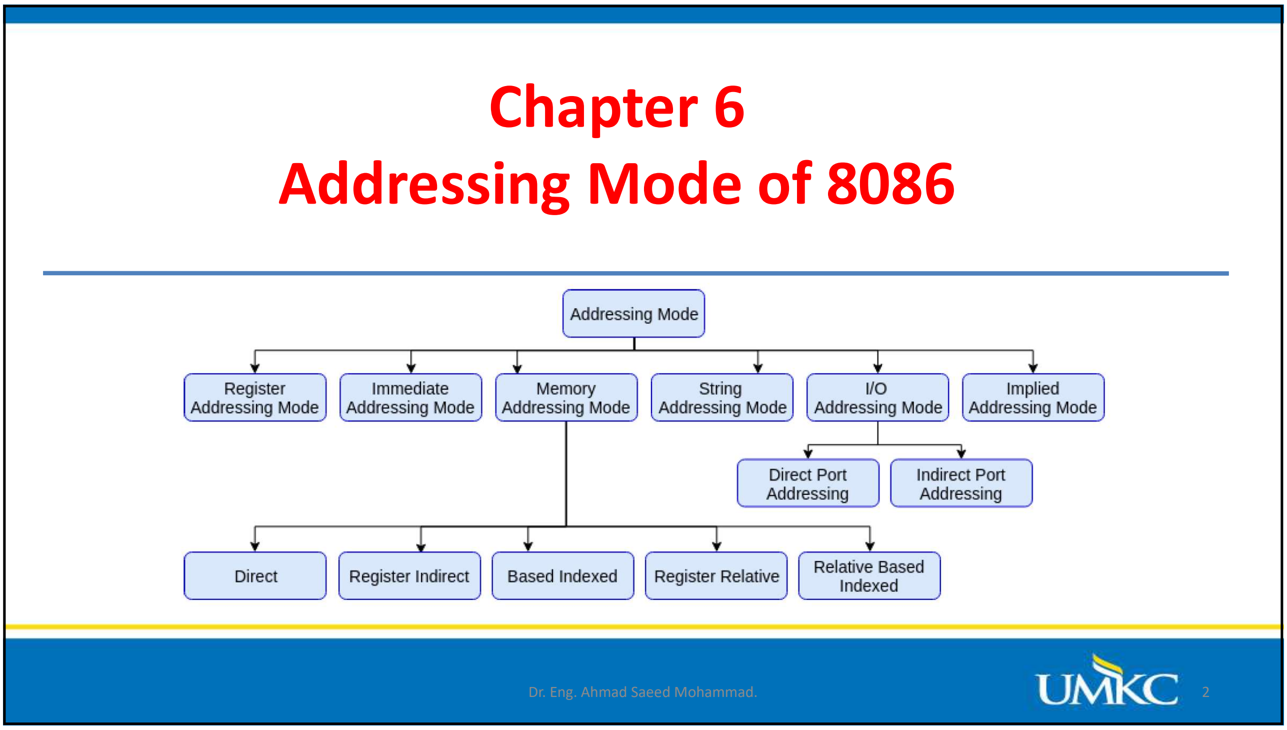
System Programming with Microprocessors

Dr. Ahmad Saeed Mohammad
Ph.D. Electrical and Computer Engineering

Dr. Eng. Ahmad Saeed Mohammad.

UMKC 1

1



Chapter 6
Addressing Mode of 8086

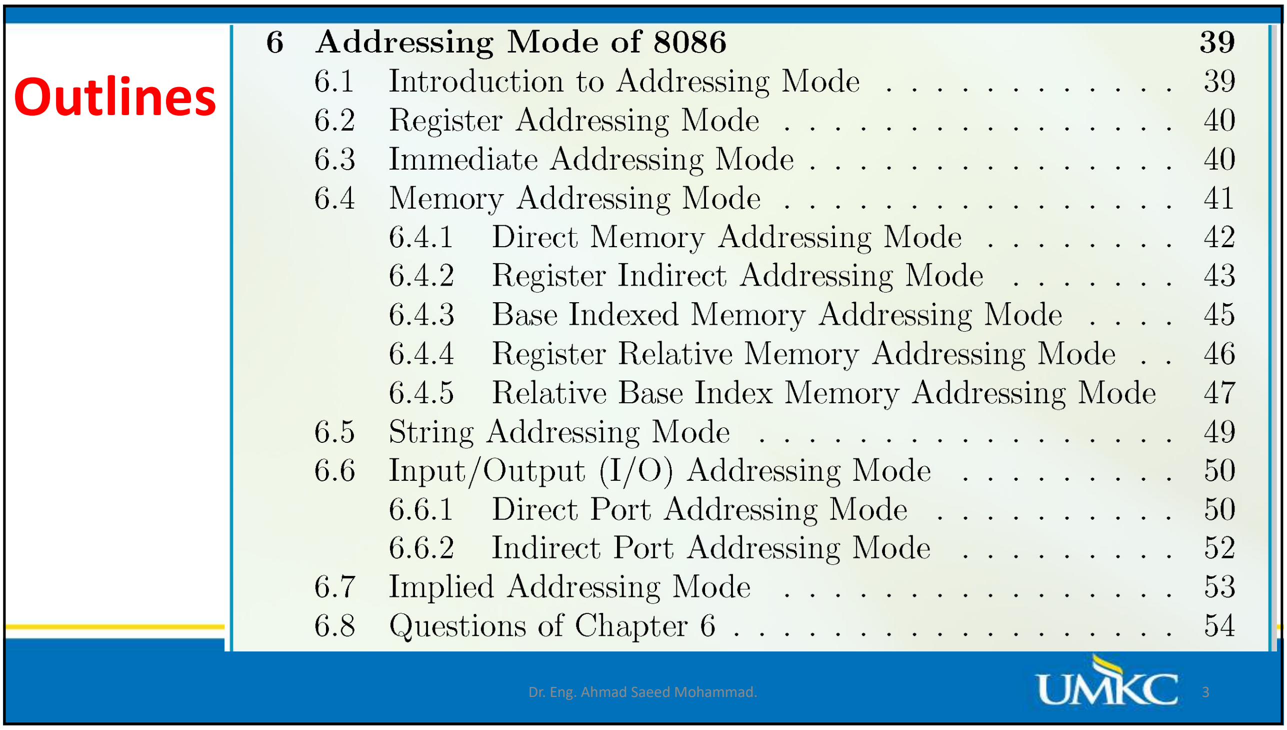
Addressing Mode

- Register Addressing Mode
- Immediate Addressing Mode
- Memory Addressing Mode
  - Direct
  - Register Indirect
  - Based Indexed
  - Register Relative
  - Relative Based Indexed
- String Addressing Mode
- I/O Addressing Mode
  - Direct Port Addressing
  - Indirect Port Addressing
- Implied Addressing Mode

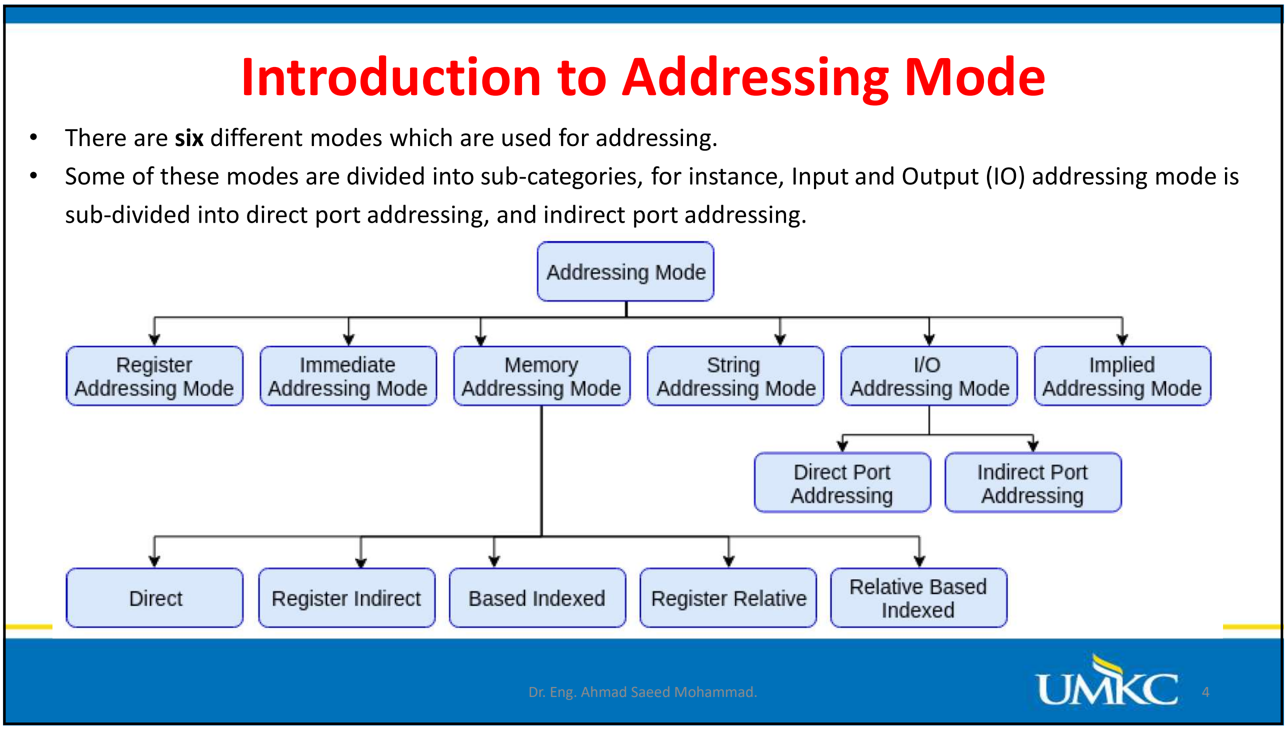
Dr. Eng. Ahmad Saeed Mohammad.

UMKC 2

2



Outlines



Introduction to Addressing Mode

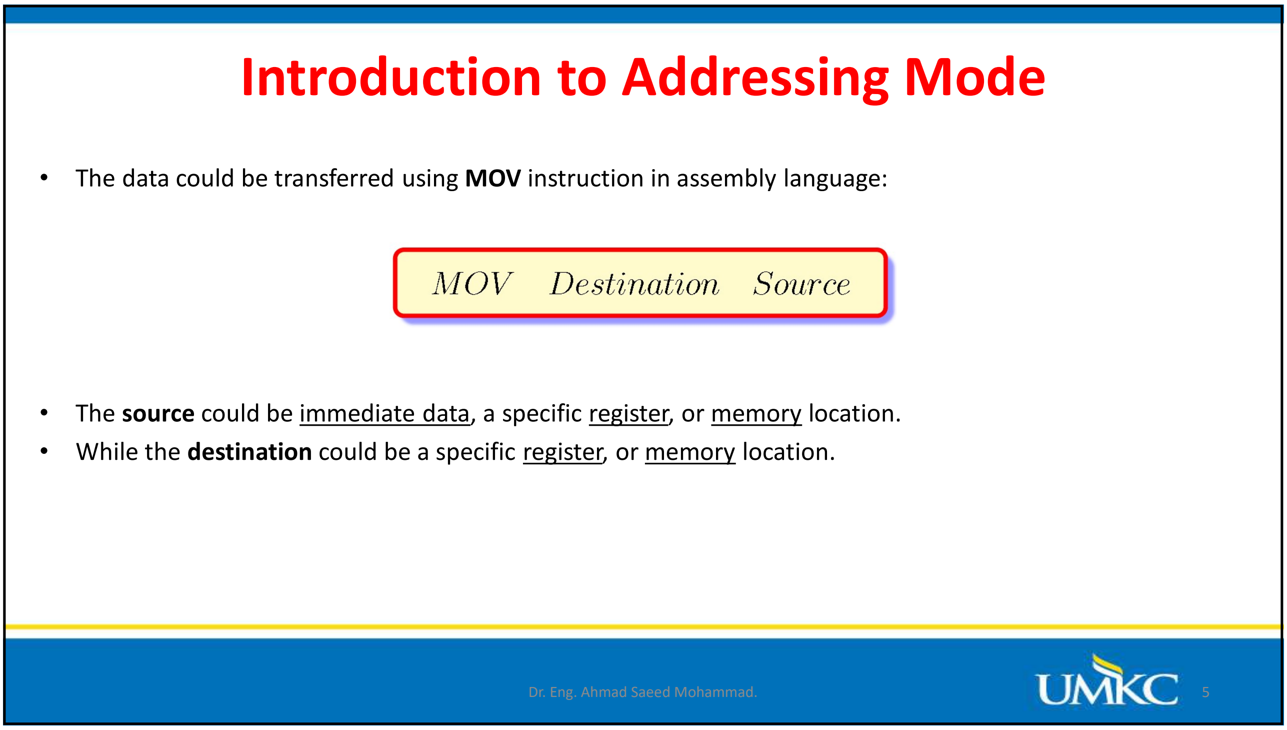
- There are **six** different modes which are used for addressing.
- Some of these modes are divided into sub-categories, for instance, Input and Output (IO) addressing mode is sub-divided into direct port addressing, and indirect port addressing.

Addressing Mode

Register Addressing Mode
Immediate Addressing Mode
Memory Addressing Mode
String Addressing Mode
I/O Addressing Mode
Implied Addressing Mode

Direct Port Addressing
Indirect Port Addressing

Direct
Register Indirect
Based Indexed
Register Relative
Relative Based Indexed

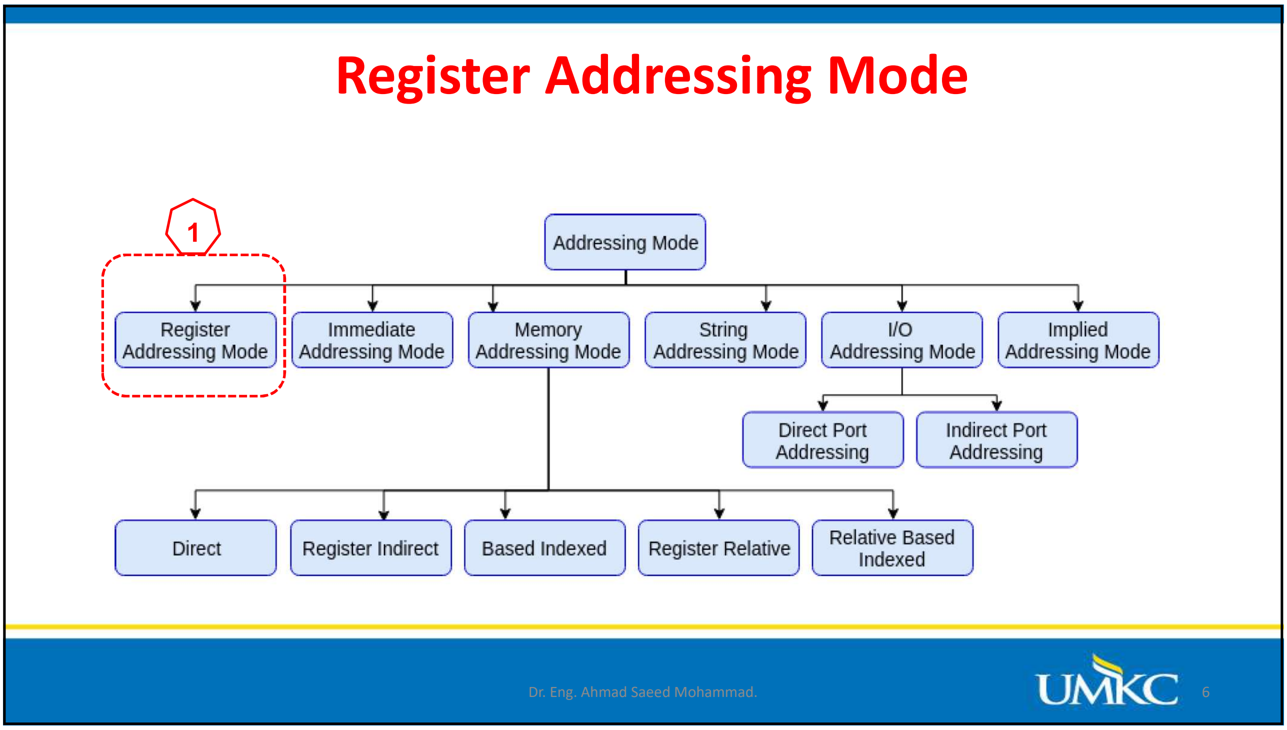


Introduction to Addressing Mode

- The data could be transferred using **MOV** instruction in assembly language:

$$MOV \quad Destination \quad Source$$

- The **source** could be immediate data, a specific register, or memory location.
- While the **destination** could be a specific register, or memory location.



Register Addressing Mode

1

Addressing Mode

Register Addressing Mode

Immediate Addressing Mode

Memory Addressing Mode

String Addressing Mode

I/O Addressing Mode

Implied Addressing Mode

Direct Port Addressing

Indirect Port Addressing

Direct

Register Indirect

Based Indexed

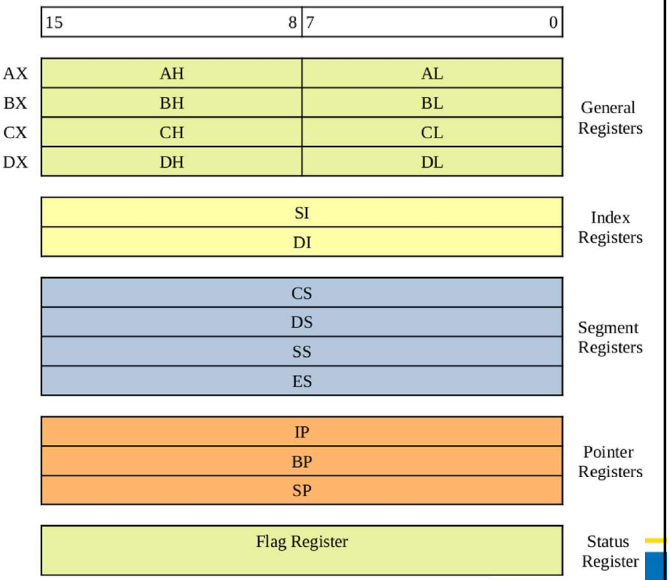
Register Relative

Relative Based Indexed

Register Addressing Mode

- The **source and destination** of the date are stored **in** a specific **register**.
- The instruction (**MOV**) will refer to a particular **register** to be copied into another **register**.
- All registers could be used except **IP** register.

$$Reg \leftarrow Reg$$



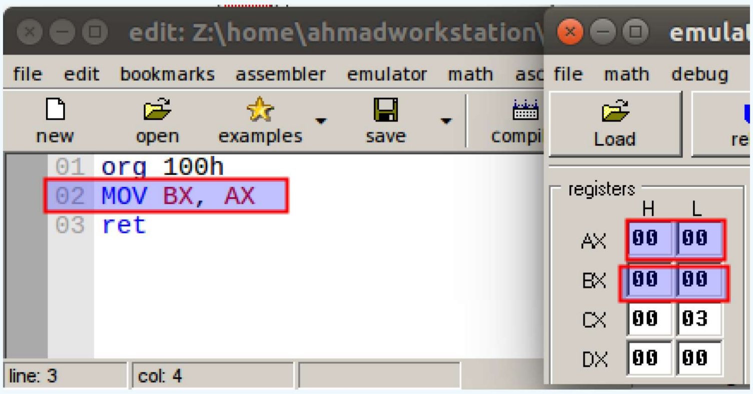
	15 8	7 0	
AX	AH	AL	General Registers
BX	BH	BL	
CX	CH	CL	
DX	DH	DL	
	SI		Index Registers
	DI		
	CS		Segment Registers
	DS		
	SS		
	ES		
	IP		Pointer Registers
	BP		
	SP		
	Flag Register		Status Register

Register Addressing Mode

Example 6.1:

Write an assembly code to transfer a data from accumulator (AX) register to base (BX) register.

Solution:

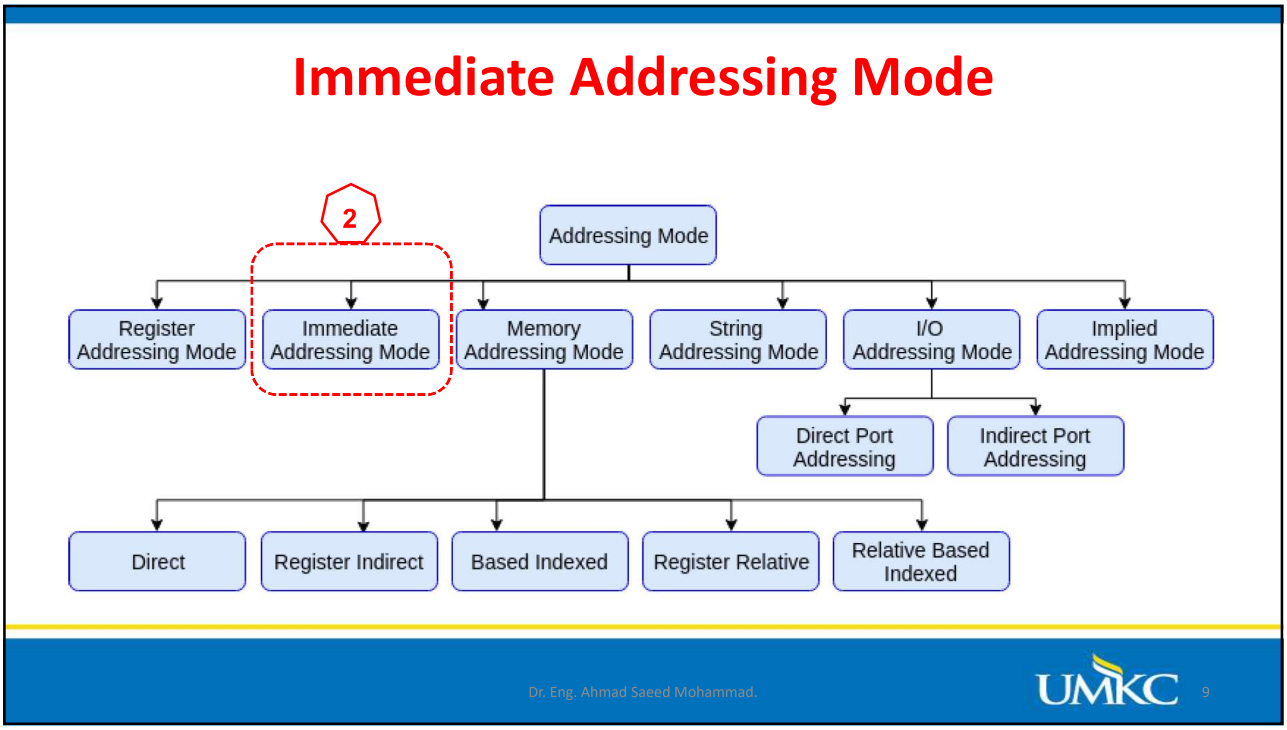


```
org 100h
MOV BX, AX
ret
```

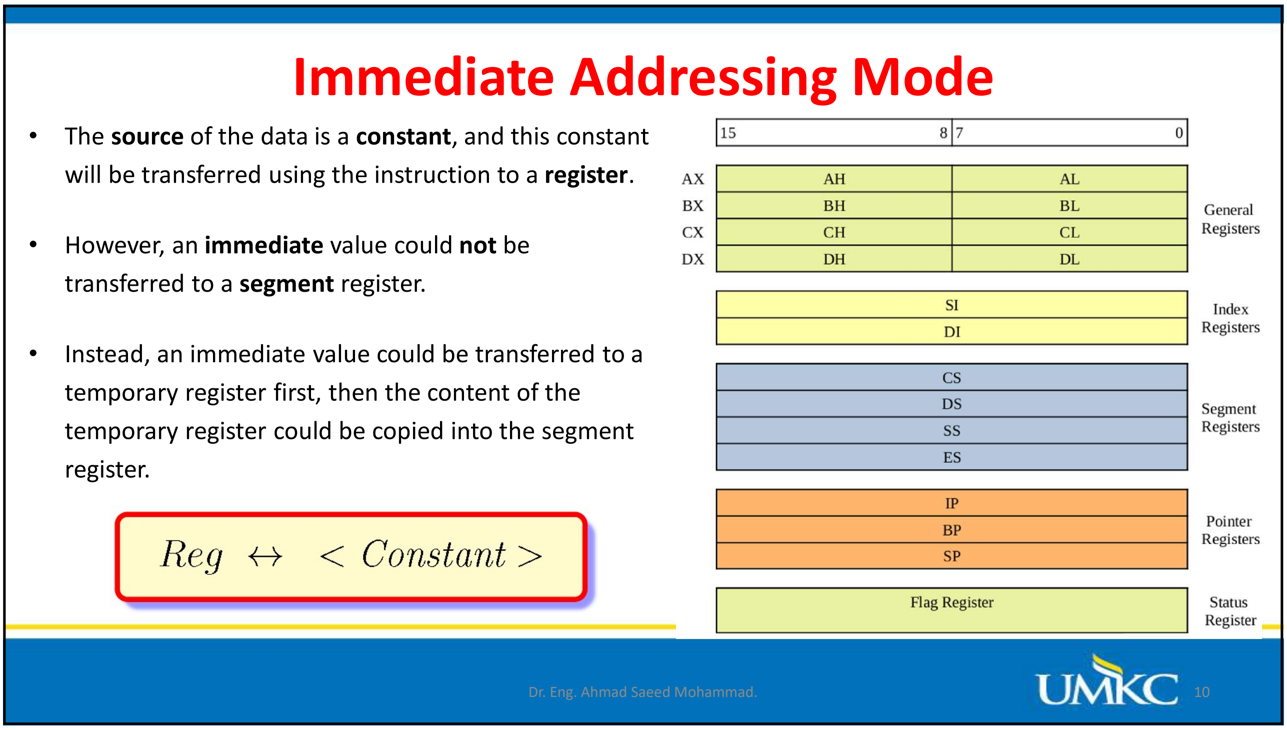
line: 3 col: 4

registers

	H	L
AX	00	00
BX	00	00
CX	00	03
DX	00	00



Immediate Addressing Mode



Immediate Addressing Mode

- The **source** of the data is a **constant**, and this constant will be transferred using the instruction to a **register**.
- However, an **immediate** value could **not** be transferred to a **segment** register.
- Instead, an immediate value could be transferred to a temporary register first, then the content of the temporary register could be copied into the segment register.

$$Reg \leftrightarrow < Constant >$$

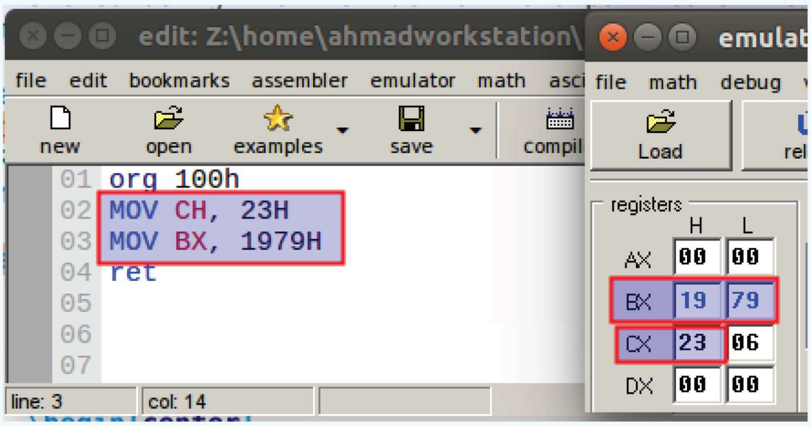
	15 — 8	7 — 0	
AX	AH	AL	General Registers
BX	BH	BL	
CX	CH	CL	
DX	DH	DL	
	SI		Index Registers
	DI		
	CS		Segment Registers
	DS		
	SS		
	ES		
	IP		Pointer Registers
	BP		
	SP		
	Flag Register		Status Register

Immediate Addressing Mode

Example 6.2:

Write an assembly code to transfer a 23 H to lower part of counter register. Also, transfer a 1979 H to base (BS) register.

Solution:

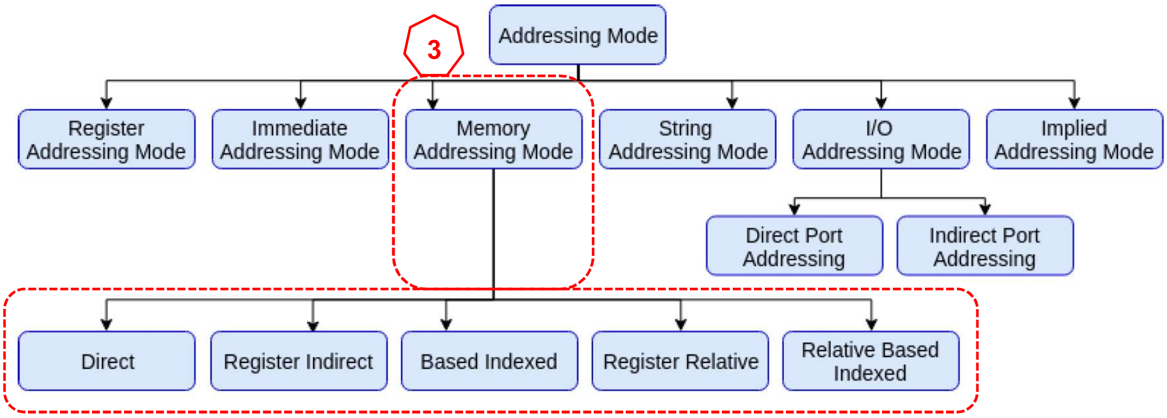

edit: Z:\home\ahmadworkstation\
file edit bookmarks assembler emulator math asci
new open examples save compil
emulat
file math debug
Load
rel
registers

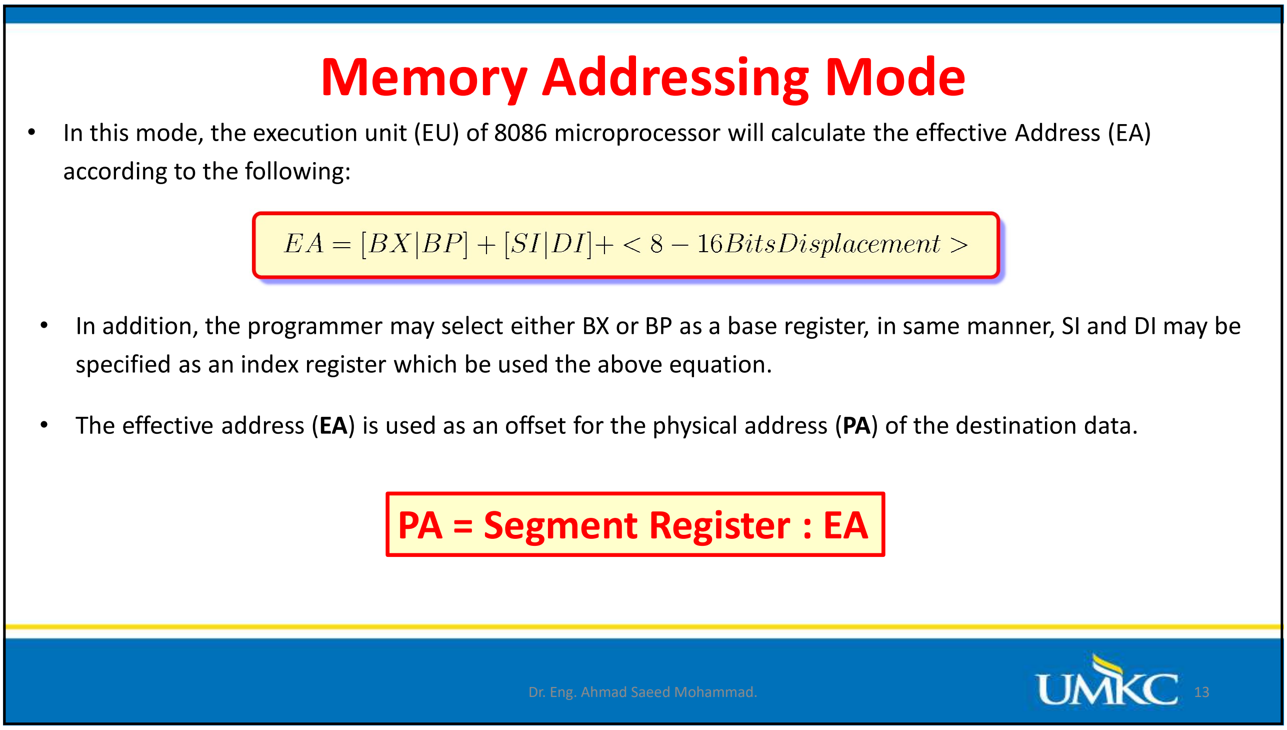
```
org 100h
MOV CH, 23H
MOV BX, 1979H
ret
```

line: 3 col: 14

	H	L
AX	00	00
BX	19	79
CX	23	06
DX	00	00

Memory Addressing Mode





Memory Addressing Mode

- In this mode, the execution unit (EU) of 8086 microprocessor will calculate the effective Address (EA) according to the following:

$$EA = [BX|BP] + [SI|DI] + < 8 - 16 Bits Displacement >$$

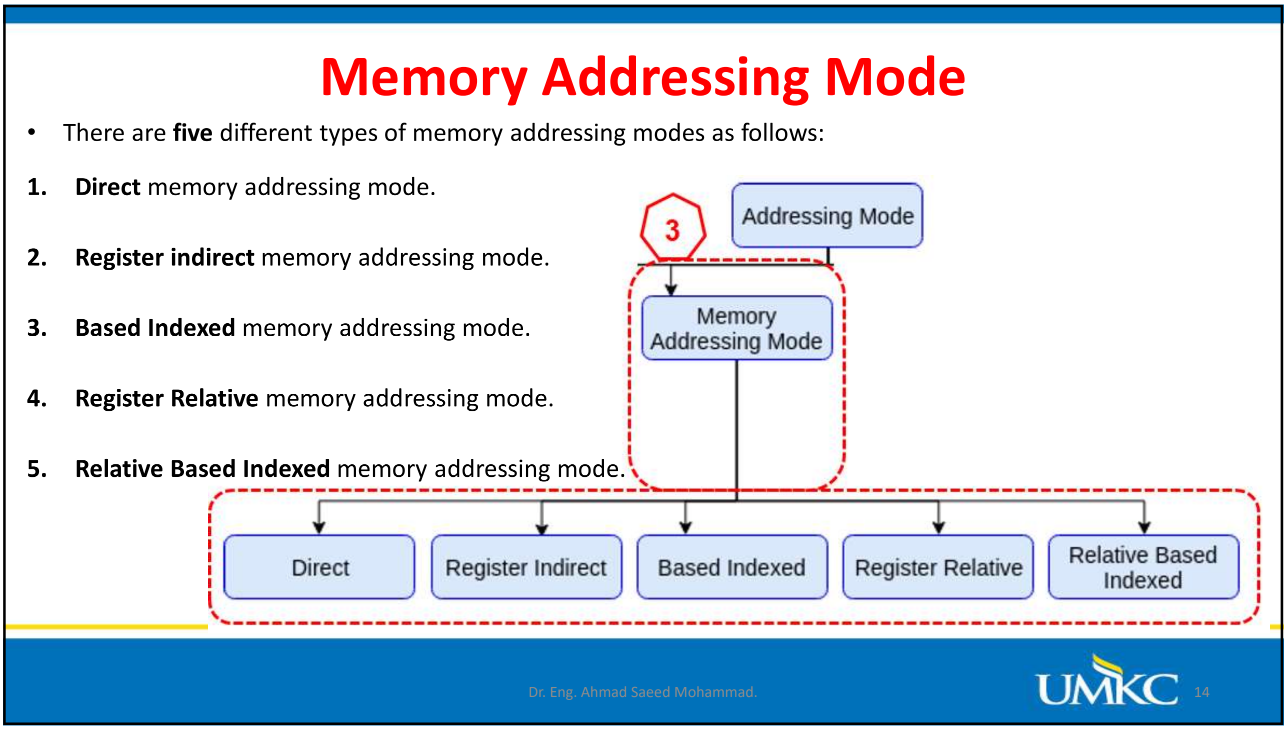
- In addition, the programmer may select either BX or BP as a base register, in same manner, SI and DI may be specified as an index register which be used the above equation.
- The effective address (**EA**) is used as an offset for the physical address (**PA**) of the destination data.

PA = Segment Register : EA

Dr. Eng. Ahmad Saeed Mohammad.

UMKC 13

13



Memory Addressing Mode

- There are **five** different types of memory addressing modes as follows:

1. **Direct** memory addressing mode.
2. **Register indirect** memory addressing mode.
3. **Based Indexed** memory addressing mode.
4. **Register Relative** memory addressing mode.
5. **Relative Based Indexed** memory addressing mode.

Dr. Eng. Ahmad Saeed Mohammad.

UMKC 14

14

Direct Memory Addressing Mode

- In this mode, EA will be implied directly in the instruction. So, there is no calculation will be involved.
- Also, a data will be copied between the memory and a register specified by the instruction.

$$\boxed{Reg \leftrightarrow [\ EA\ ]}$$

- The content of memory is either a byte (8 bits) or word (16 bits), and the memory location (physical address) is calculated as the following:

$$\boxed{Physical_{Address} = [Segment_{Address}] \times 10]_H + \text{Offset}_{Address}}$$

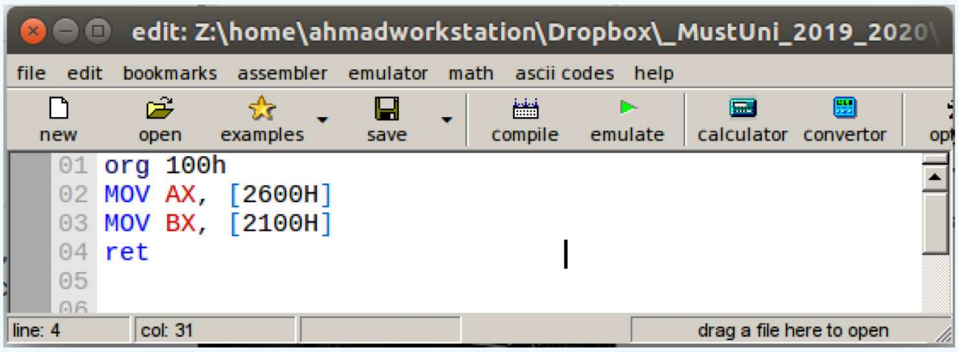
- For instance: **PA = DS : EA**

Direct Memory Addressing Mode

Example 6.3:

Write an assembly code to transfer the memory content of address 2600 H in the data segment to the accumulator. Also, transfer the memory content of address 2100 H to the base register.

Solution:

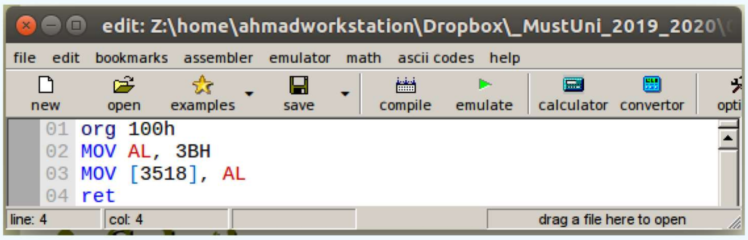

edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new
open
examples
save
compile
emulate
calculator
convertor
opt
line: 4
col: 31
drag a file here to open

```
org 100h
MOV AX, [2600H]
MOV BX, [2100H]
ret
```

Direct Memory Addressing Mode

Example 6.4:

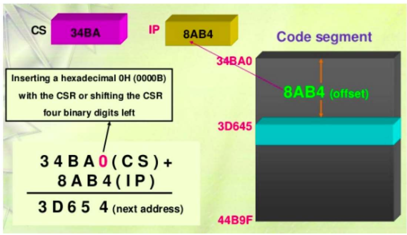
Find the physical address of the memory location and its contents after the execution of the following, assuming that DS = 1512H.


edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new
open
examples
save
compile
emulate
calculator
convertor
opti
line: 4
col: 4
drag a file here to open

```
01 org 100h
02 MOV AL, 3BH
03 MOV [3518], AL
04 ret
```

Solution:

- Line 2: $3B$ H is copied into AL register.
- Line 3: The content of AL copied into memory address DS : 3518 which is 1512 : 3518
- Shift DS into left 15120, then add it to 3518, Thus 15120 H + 3518 H = 18638 H

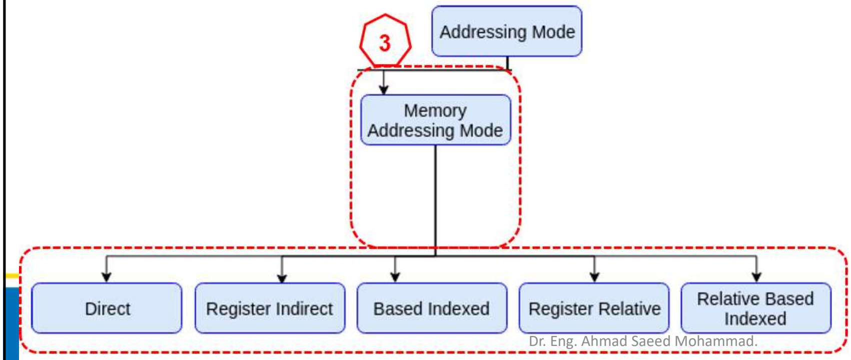

CS
34BA
IP
8AB4
Code segment
34BA0
Inserting a hexadecimal 0H (0000B) with the CSR or shifting the CSR four binary digits left
8AB4 (offset)
3D645
34BA0(CS)+
8AB4(IP)
3D65 4 (next address)
44B9F

$$Physical_{Address} = [Segment_{Address}] \times 10]_H + \text{Offset}_{Address}$$

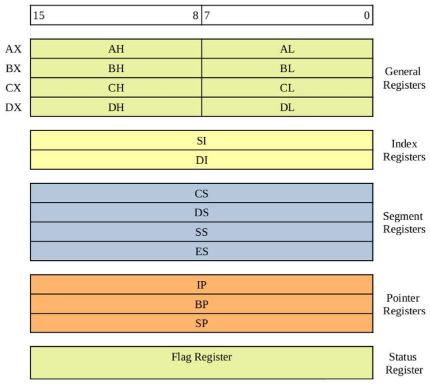
Register Indirect Addressing Mode

- A byte **8 bits** or a word **16 bits** will be transferred between a register and a memory location indexed by an index (**SI, DI**) or base (**BX**) register.

$$Reg \leftrightarrow [DS \ : \ \{SI \mid DI \mid BX\}]$$



Dr. Eng. Ahmad Saeed Mohammad.

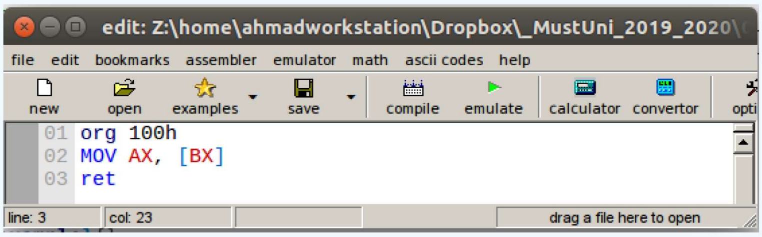


	15 8	7 0	
AX	AH	AL	General Registers
BX	BH	BL	
CX	CH	CL	
DX	DH	DL	
	SI		Index Registers
	DI		
	CS		Segment Registers
	DS		
	SS		
	ES		
	IP		Pointer Registers
	BP		
	SP		
	Flag Register		Status Register

Register Indirect Addressing Mode

Example 6.5:

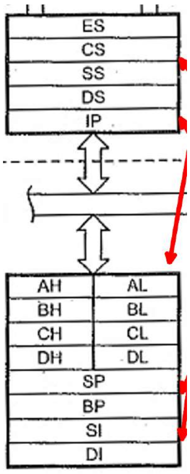
Find the physical address of the memory location after the execution of the following, assuming that DS= 1000H, and BX= 1234 H.


edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor opti
line: 3 col: 23 drag a file here to open

```
01 org 100h
02 MOV AX, [BX]
03 ret
```

Solution:

- Line 2: The content of memory location pointed by $DS : BX$ is copied into AX register.
- The physical address is calculated as:
  $DS : BX = 1000\ H : 1234\ H$
  $= 10000\ H + 1234\ H = 11234\ H$

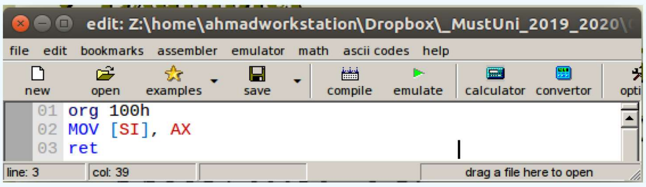


$Reg \leftrightarrow [DS\ :\ \{SI \mid DI \mid BX\}]$

Register Indirect Addressing Mode

Example 6.6:

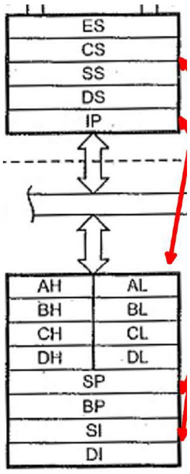
Assume that DS $= 1120$ H, SI $= 2498$ H, and AX $= 17FE$ H Show the contents of memory locations after the execution of


edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor opti
line: 3 col: 39 drag a file here to open

```
01 org 100h
02 MOV [SI], AX
03 ret
```

Solution:

- Line 2: The content of AX register which is $17FE$ H is copied into the memory location $DS : SI$, and $DS : SI{+}1$.
- The physical address is calculated as:
  $DS : SI = 1120\ H : 2498\ H = 11200\ H + 2498\ H = 13698\ H$
  $DS : SI + 1 = 1120\ H : 2498 + 1\ H = 11200\ H + 2499\ H = 13699\ H$
- The memory content:
  $13698\ H$ will hold FE H :: AL : Low byte
  $13699\ H$ will hold 17 H :: AH : High byte

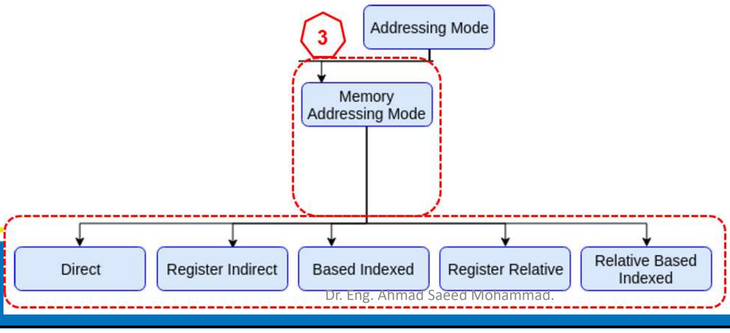


$Reg \leftrightarrow [DS\ :\ \{SI \mid DI \mid BX\}]$

Base Indexed Memory Addressing Mode

- A byte **8 bits** or a word **16 bits** will be transferred between a **register** and a **memory** location indexed by an index (**SI** or **DI**) **plus** base (**BX** or **BP**) register.
- This mode is considered as a combination between based mode and indexed mode in which one base register and one index register is used at each time.

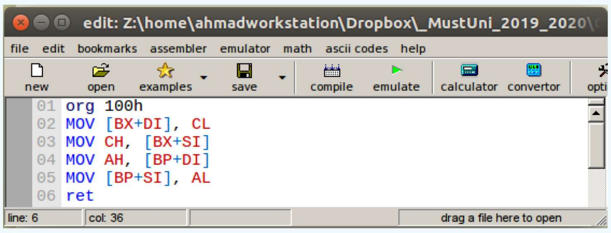
$$Reg \leftrightarrow [\{DS \mid SS \mid ES\} \ : \ \{SI \mid DI\} \ + \ \{BX \mid BP\}]$$



Base Indexed Memory Addressing Mode

Example 6.7:

Describable each line and write the equations of the physical address for the following assembly code:


edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor opti
line: 6 col: 36 drag a file here to open

```
01 org 100h
02 MOV [BX+DI], CL
03 MOV CH, [BX+SI]
04 MOV AH, [BP+DI]
05 MOV [BP+SI], AL
06 ret
```

Solution:

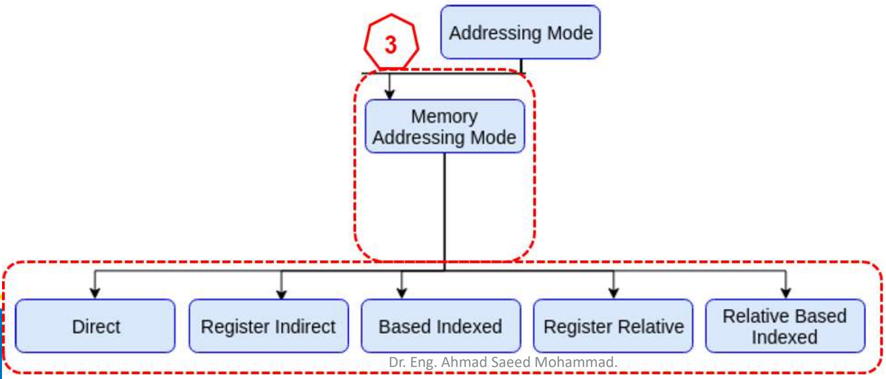
- **Line 2**: Copy contents of CL into $[DS : BX + DI]$:
  $PA \ = \ DS \ \times \ 10 \ + \ BX \ + \ DI$
- **Line 3**: Copy contents of the $[DS : BX + SI]$ into CH:
  $PA \ = \ DS \ \times \ 10 \ + \ BX \ + \ SI$
- **Line 4**: Copy contents of the $[SS : BP + SI]$ into CH:
  $PA \ = \ SS \ \times \ 10 \ + \ BP \ + \ SI$
- **Line 5**: Copy contents of AL into $[SS : BP + SI]$:
  $PA \ = \ SS \ \times \ 10 \ + \ BX \ + \ DI$

$$Physical_{Address} = [Segment_{Address}] \times 10]_H + \text{Offset}_{Address}$$

Register Relative Memory Addressing Mode

- A byte **8 bits** or a word **16 bits** will be transferred between a **register** and a **memory** location **indexed by** an index (**SI**, **DI**) or base (**BX**) register **plus displacement**.

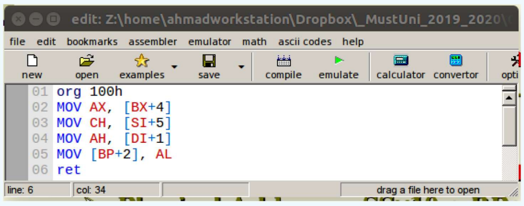
$$Reg \leftrightarrow [\{DS \mid SS \mid ES\} \;:\; \{SI \mid DI\} \;\mid\; \{BX \mid BP\} \;+\; Disp.]$$



Register Relative Memory Addressing Mode

Example 6.8:

Describable each line and write the equations of the physical address for the following assembly code:


edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor opti
line: 6
col: 34
drag a file here to open

```
01 org 100h
02 MOV AX, [BX+4]
03 MOV CH, [SI+5]
04 MOV AH, [DI+1]
05 MOV [BP+2], AL
06 ret
```

Solution:

- **Line 2**: Copy contents of the $[DS : BX + 4]$ into AX:
  $PA = DS \times 10 + BX + 4$
- **Line 3**: Copy contents of the $[DS : SI + 5]$ into CH:
  $PA = DS \times 10 + SI + 5$
- **Line 4**: Copy contents of the $[DS : DI + 1]$ into AH:
  $PA = DS \times 10 + DI + SI$
- **Line 5**: Copy contents of AL into $[SS : BP + 2]$:
  $PA = SS \times 10 + BP + 2$

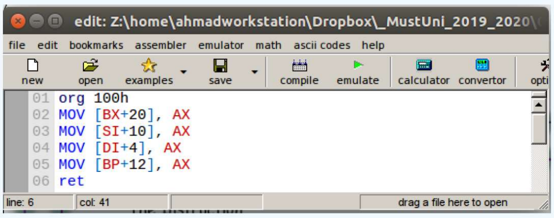
$$Reg \leftrightarrow [\{DS \mid SS \mid ES\} \;:\; \{SI \mid DI\} \;\mid\; \{BX \mid BP\} \;+\; Disp.]$$

$$Physical_{Address} = [Segment_{Address}] \times 10]_H + \text{Offset}_{Address}$$

Register Relative Memory Addressing Mode

Example 6.9:

Calculate the physical address for the following assembly code, assume that DS = 4500 H, SS = 2000 H, BX = 2100 H, SI = 1486 H, DI = 8500 H, BP= 7814 H, and AX = 2512 H:


edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor opti
line: 6 col: 41 drag a file here to open

```
01 org 100h
02 MOV [BX+20], AX
03 MOV [SI+10], AX
04 MOV [DI+4], AX
05 MOV [BP+12], AX
06 ret
```

Solution: Since: $PA = Seg.Reg. \times 10 + offsetReg. + Disp.$

- **Line 2**: $PA = 45000 + 2100 + 20 = 47120$:
  Location $47120 = (12)$ and $47121 = (25)$
- **Line 3**: $PA = 45000 + 1486 + 10 = 46496$:
  Location $46496 = (12)$ and $46497 = (25)$
- **Line 4**: $PA = 45000 + 8500 + 4 = 4D504$:
  Location $4D504 = (12)$ and $4D505 = (25)$
- **Line 5**: $PA = 20000 + 7814 + 12 = 27826$:
  Location $27826 = (12)$ and $27827 = (25)$

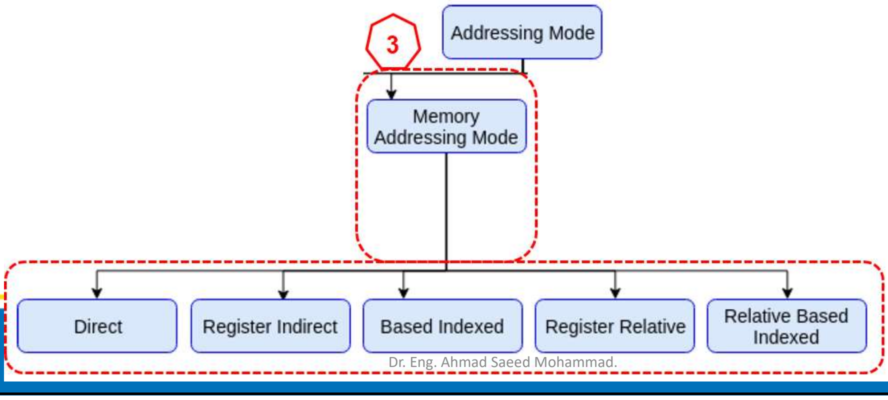
$$Reg \leftrightarrow [\{DS \mid SS \mid ES\} \ : \ \{SI \mid DI\} \ \mid \ \{BX \mid BP\} \ + \ Disp.]$$

$$Physical_{Address} = [Segment_{Address}] \times 10]_H + \text{Offset}_{Address}$$

Relative Base Index Memory Addressing Mode

- This mode is similar to the base indexed addressing mode**, in addition, it adds a displacement** besides the base and index register.

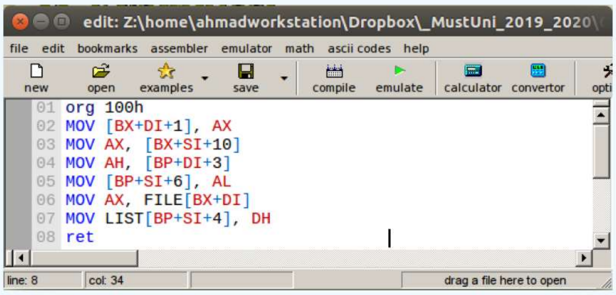
$$Reg \leftrightarrow [\{DS \mid SS \mid ES\} \ : \ \{SI \mid DI\} \ + \ \{BX \mid BP\} \ + \ Disp.]$$



Relative Base Index Memory Addressing Mode

Example 6.10:

Describable each line and write the equations of the physical address for the following assembly code:



edit: Z:\home\ahmadworkstation\Dropbox_MustUni_2019_2020\
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor opti
line: 8
col: 34
drag a file here to open

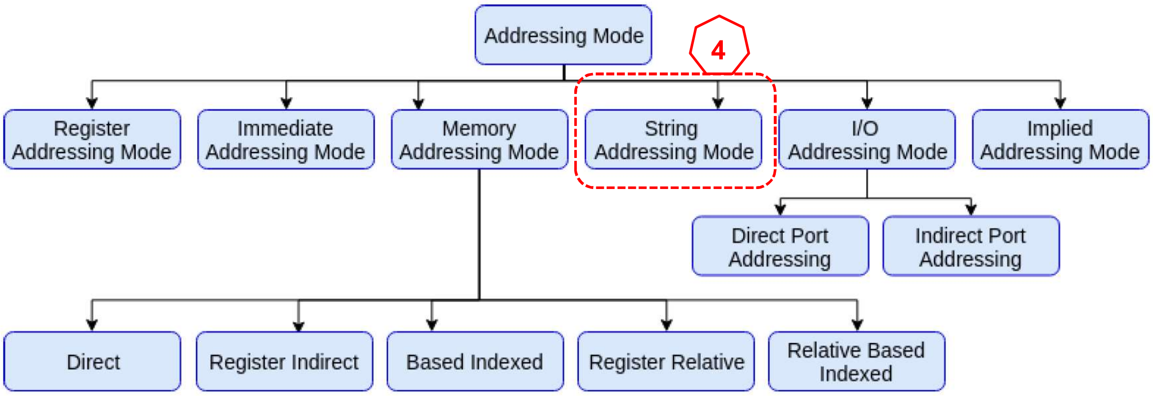
```
01 org 100h
02 MOV [BX+DI+1], AX
03 MOV AX, [BX+SI+10]
04 MOV AH, [BP+DI+3]
05 MOV [BP+SI+6], AL
06 MOV AX, FILE[BX+DI]
07 MOV LIST[BP+SI+4], DH
08 ret
```

Solution:

- **Line 2**: Copy contents of AX into $[DS : BX + DI + 1]$: $PA = DS \times 10 + (BX + DI + 1)$
- **Line 3**: Copy contents of the $[DS : BX + SI + 10]$ into AX: $PA = DS \times 10 + (BX + SI + 10)$
- **Line 4**: Copy contents of the $[SS : BP + DI + 3]$ into AH: $PA = SS \times 10 + (BP + DI + 3)$
- **Line 5**: Copy contents of AL into $[SS : BP + SI + 6]$: $PA = SS \times 10 + BP + SI + 6$
- **Line 6**: Copy contents of the $[DS : BX + DI + FILE]$ into AX: $PA = DS \times 10 + (BP + DI + FILE)$
- **Line 7**: Copy contents of the $[SS : BP + SI + 4 + LIST]$ into DH: $PA = SS \times 10 + (BP + DI + 4 + LIST)$

$$\boxed{Physical_{Address} = [Segment_{Address}] \times 10]_H + \text{Offset}_{Address}}$$

String Addressing Mode



String Addressing Mode

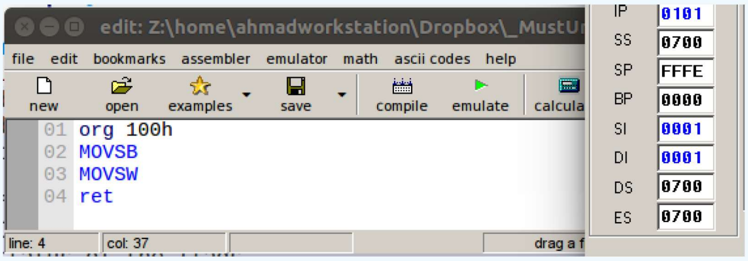
- This addressing mode is used when a string instruction is executed.
- Neither **SI** or **DI** register will appear in instruction code; however, **SI** will point to the first byte or word of the **source** data string, and **DI** will point to the first byte or word of the **destination** data string.
- Also, **SI** and **DI** are incremented or decremented according to the status of the flags.

$$\boxed{ES : [DI] \leftarrow DS : [SI]}$$

String Addressing Mode

Example 6.11:

Describe the following assembly code, and which addressing mode is used, also mentioned which flag is used to increment/decrement the address of data and how to clear this flag.


edit: Z:\home\ahmadworkstation\Dropbox_MustU
file edit bookmarks assembler emulator math ascii codes help
new
open
examples
save
compile
emulate
calcula
drag a f

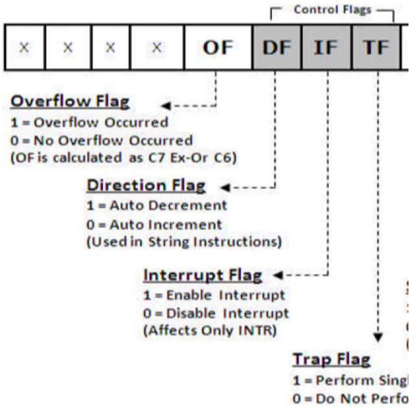
```
01 org 100h
02 MOVSB
03 MOVSW
04 ret
```

line: 4 col: 37

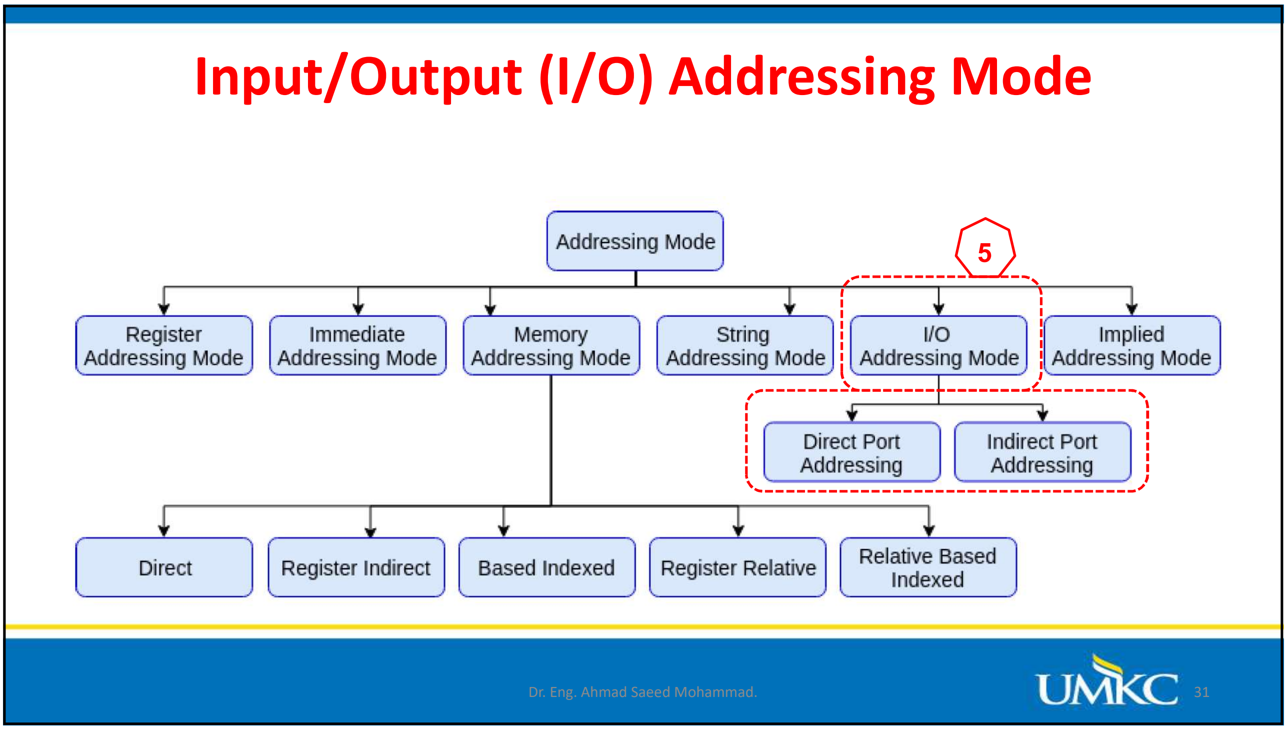
IP 0101
SS 0700
SP FFFE
BP 0000
SI 0001
DI 0001
DS 0700
ES 0700

Solution:

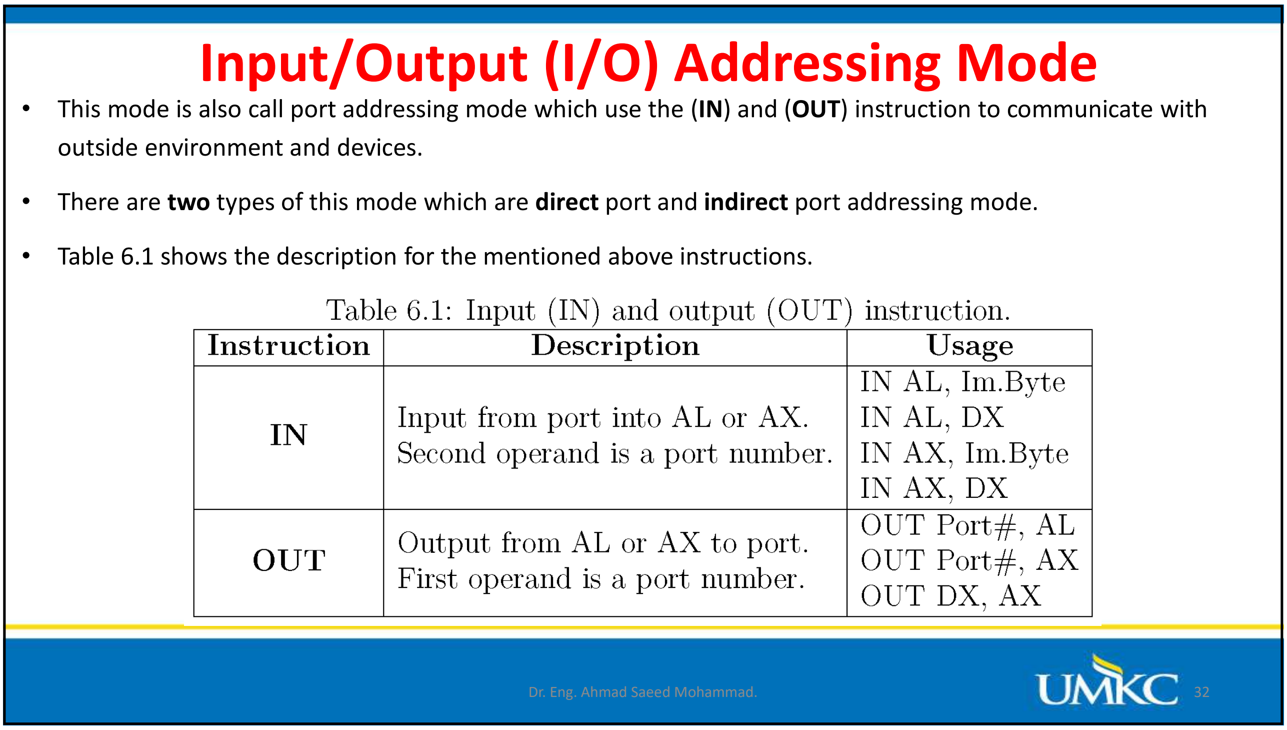
Line 2: Copy byte at DS:[SI] to ES:[DI]. Then, Update SI and DI.
Line 3: Copy word at DS:[SI] to ES:[DI]. Then, Update SI and DI.
This mode is string addressing mode. Also, IF DF flag is (0), then both SI and DI will incremented. The instruction (CLD) will clear the Direction Flag (DF).



$$\boxed{ES : [DI] \leftarrow DS : [SI]}$$



Input/Output (I/O) Addressing Mode



Input/Output (I/O) Addressing Mode

- This mode is also call port addressing mode which use the (**IN**) and (**OUT**) instruction to communicate with outside environment and devices.
- There are **two** types of this mode which are **direct** port and **indirect** port addressing mode.
- Table 6.1 shows the description for the mentioned above instructions.

Table 6.1: Input (IN) and output (OUT) instruction.

Instruction	Description	Usage
IN	Input from port into AL or AX. Second operand is a port number.	IN AL, Im.Byte IN AL, DX IN AX, Im.Byte IN AX, DX
OUT	Output from AL or AX to port. First operand is a port number.	OUT Port#, AL OUT Port#, AX OUT DX, AX

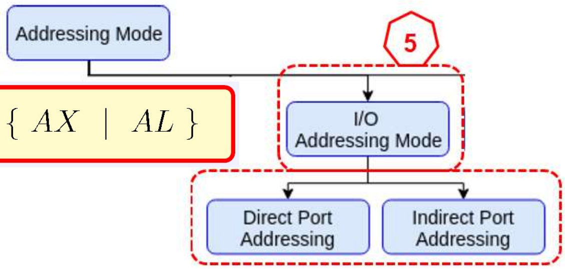
Direct Port Addressing Mode

- For **direct input operation**, the data will be applied directly to destination register (AL or AX) using (IN) instruction as shown in the following equation:

$$\boxed{Direct\ \ Mode \Rightarrow IN :: \{\ AX\ \ |\ \ AL\ \} \leftarrow \{PortNum\ \}}$$

- For **direct output operation** a direct port number (**0** to **255**) is used as shown in the following equation:

$$\boxed{Direct\ \ Mode \Rightarrow OUT :: PortNum \leftarrow \{\ AX\ \ |\ \ AL\ \}}$$

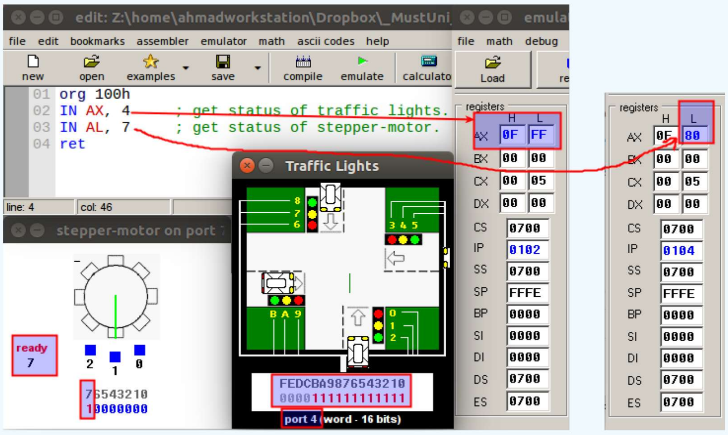


Direct Port Addressing Mode

Example 6.12:

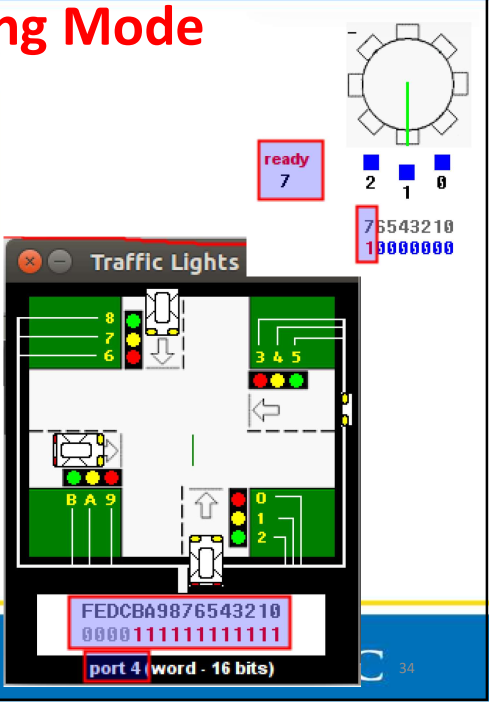
Write an assembly code to get status of traffic lights (Port number 4). Also, to get status of stepper-motor (Port number 7). Use direct addressing mode.

Solution:


edit: Z:\home\ahmadworkstation\Dropbox_MustUni
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculato
emula
file math debug
Load
line: 4
col: 46
stepper-motor on port 7
Traffic Lights
registers

```
org 100h
IN AX, 4      ; get status of traffic lights.
IN AL, 7      ; get status of stepper-motor.
ret
```

```
registers          registers
     H  L               H  L
AX   0F FF         AX   0F 80
BX   00 00         BX   00 00
CX   00 05         CX   00 05
DX   00 00         DX   00 00
CS   0700          CS   0700
IP   0102          IP   0104
SS   0700          SS   0700
SP   FFFE          SP   FFFE
BP   0000          BP   0000
SI   0000          SI   0000
DI   0000          DI   0000
DS   0700          DS   0700
ES   0700          ES   0700
```



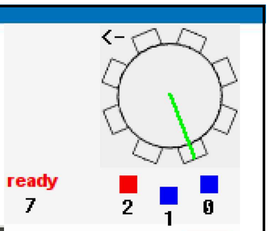
```
ready
7

76543210
10000000

FEDCBA9876543210
0000111111111111
port 4 (word - 16 bits)
```

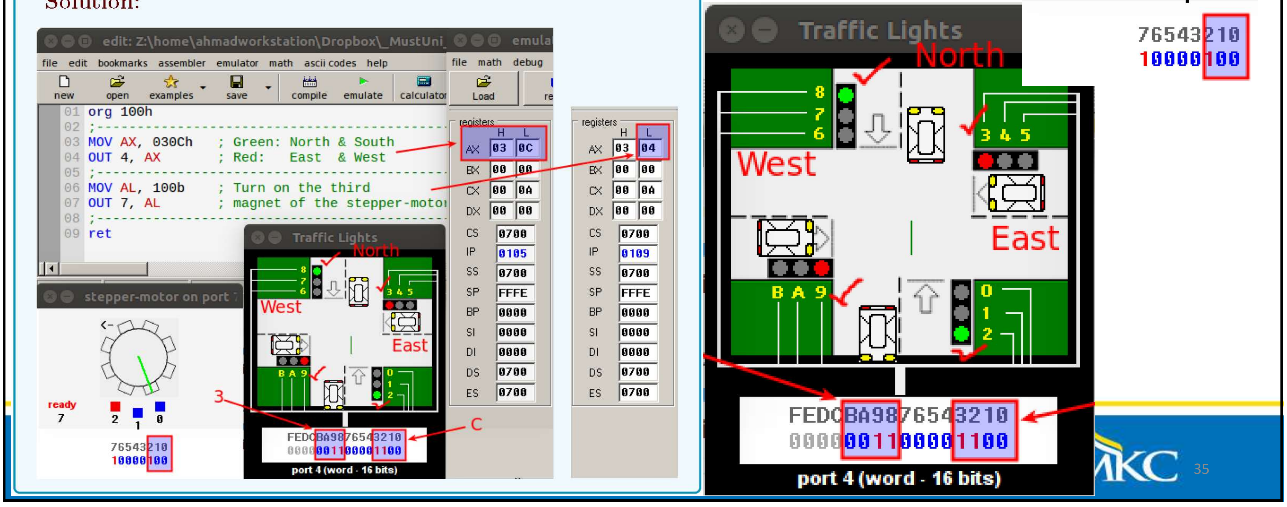
Direct Port Addressing Mode

Example 6.13:

Write an assembly code to turn the green traffic light for north and south, and turn red light for east and west. Also, turn on the third magnet of the stepper-motor. Use direct addressing mode.

Solution:





```
org 100h
;-----------------------------------------------------
MOV AX, 030Ch    ; Green: North & South
OUT 4, AX        ; Red:   East  & West
;-----------------------------------------------------
MOV AL, 100b     ; Turn on the third
OUT 7, AL        ; magnet of the stepper-motor
;-----------------------------------------------------
ret
```

registers	H	L
AX	03	0C
BX	00	00
CX	00	0A
DX	00	00
CS	0700	
IP	0105	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

registers	H	L
AX	03	04
BX	00	00
CX	00	0A
DX	00	00
CS	0700	
IP	0109	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

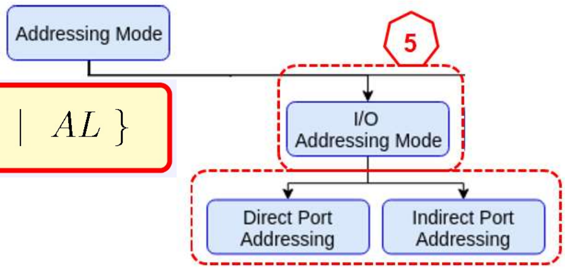
Indirect Port Addressing Mode

- For **indirect input operation**, the data will be applied indirectly using Data (**DX**) register to the destination register (**AL** or **AX**) as shown in the following equation:

$$\boxed{Indirect\ \ Mode \Rightarrow IN :: \{\ AX\ \ |\ \ AL\ \} \leftarrow \{DX\ \}}$$

- For **indirect output operation** The **DX** register used <u>instead of direct port number </u>as shown:

$$\boxed{Indirect\ \ Mode \Rightarrow OUT :: DX \leftarrow \{\ AX\ \ |\ \ AL\ \}}$$

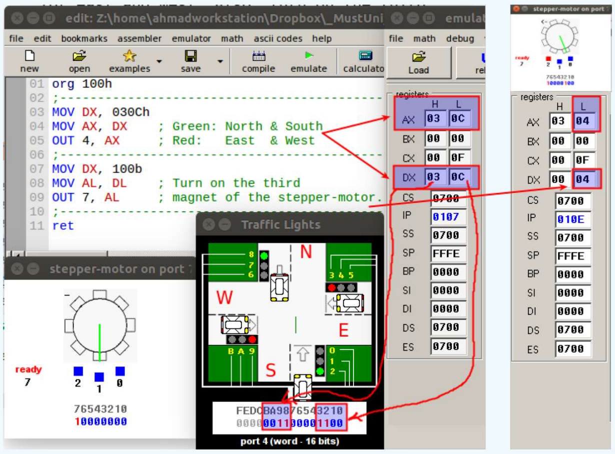


Indirect Port Addressing Mode

Example 6.14:

Write an assembly code to turn the green traffic light for north and south, and turn red light for east and west. Also, turn on the third magnet of the stepper-motor. Use indirect addressing mode.

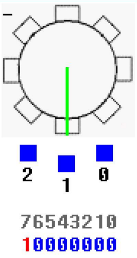
Solution:

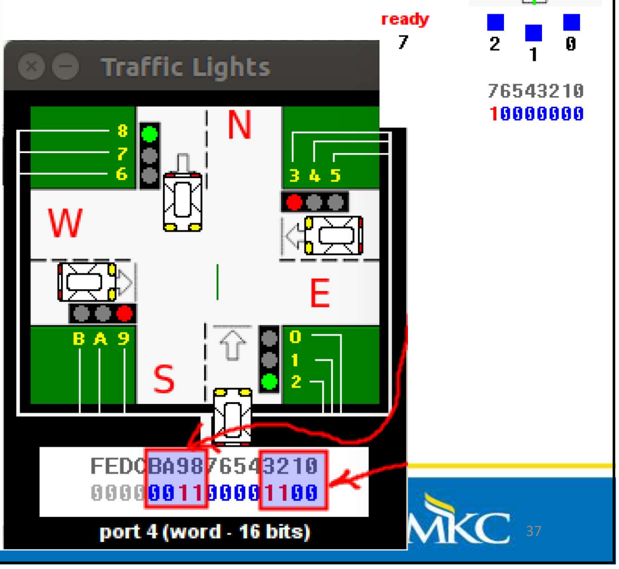


```
org 100h
;-----------------------------------------------
MOV DX, 030Ch
MOV AX, DX      ; Green: North & South
OUT 4, AX       ; Red:   East  & West
;-----------------------------------------------
MOV DX, 100b
MOV AL, DL      ; Turn on the third
OUT 7, AL       ; magnet of the stepper-motor.
;-----------------------------------------------
ret
```

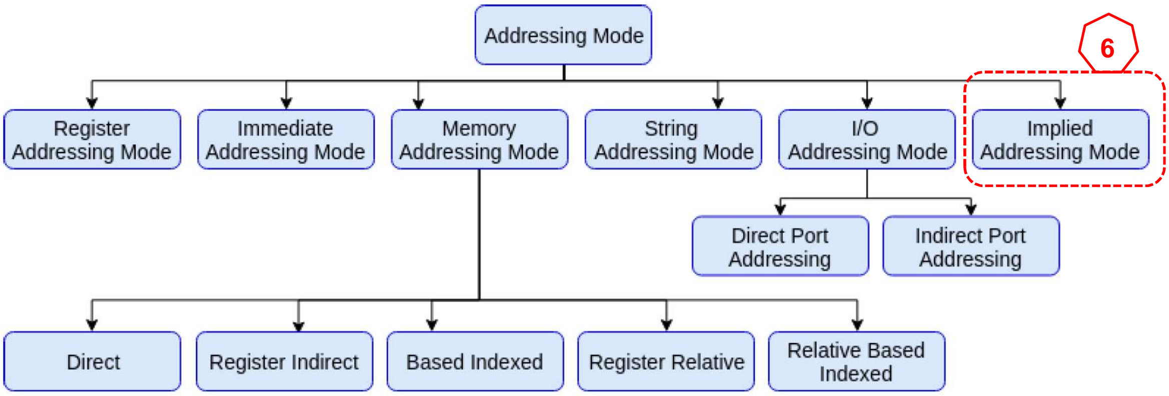
```
registers
    H  L
AX  03 0C
BX  00 00
CX  00 0F
DX  03 0C
CS  0700
IP  0107
SS  0700
SP  FFFE
BP  0000
SI  0000
DI  0000
DS  0700
ES  0700
```

edit: Z:\home\ahmadworkstation\Dropbox_MustUni
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculato
emulat
file math debug
Load
stepper-motor on port 7
Traffic Lights
port 4 (word - 16 bits)





Implied Addressing Mode



Implied Addressing Mode

- In this mode, the address will be specified by an **operand of size 8-bit** signed value which is relative (**displacement**) to the instruction pointer (**IP**).

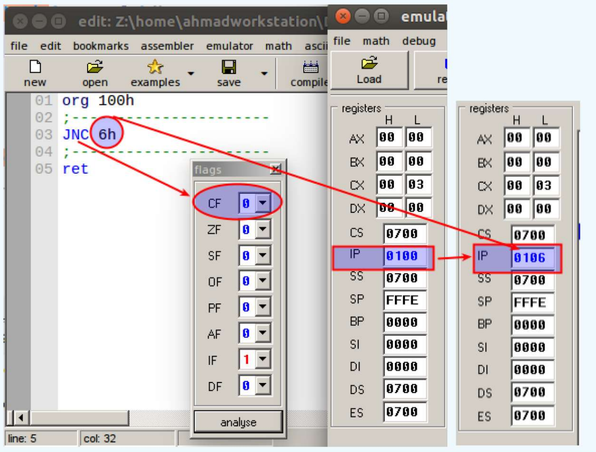
$$\boxed{[IP] \leftarrow [IP \quad + \quad Operand]}$$

Implied Addressing Mode

Example 6.15:

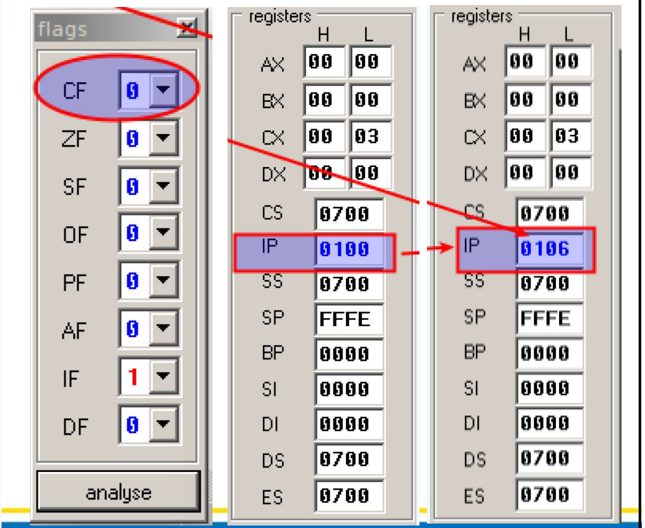
Write an assembly code to increment the instruction pointer (IP) by (6h) if carry flag is zero.

Solution:



```
org 100h
;-------------------
JNC 6h
;-------------------
ret
```

edit: Z:\home\ahmadworkstation\
file edit bookmarks assembler emulator math ascii
new open examples save compile
emula
file math debug
Load
line: 5
col: 32
analyse



flags	
CF	0
ZF	0
SF	0
OF	0
PF	0
AF	0
IF	1
DF	0

registers	H	L	registers	H	L
AX	00	00	AX	00	00
BX	00	00	BX	00	00
CX	00	03	CX	00	03
DX	00	00	DX	00	00
CS	0700		CS	0700	
IP	0100		IP	0106	
SS	0700		SS	0700	
SP	FFFE		SP	FFFE	
BP	0000		BP	0000	
SI	0000		SI	0000	
DI	0000		DI	0000	
DS	0700		DS	0700	
ES	0700		ES	0700	

Questions of Chapter 6

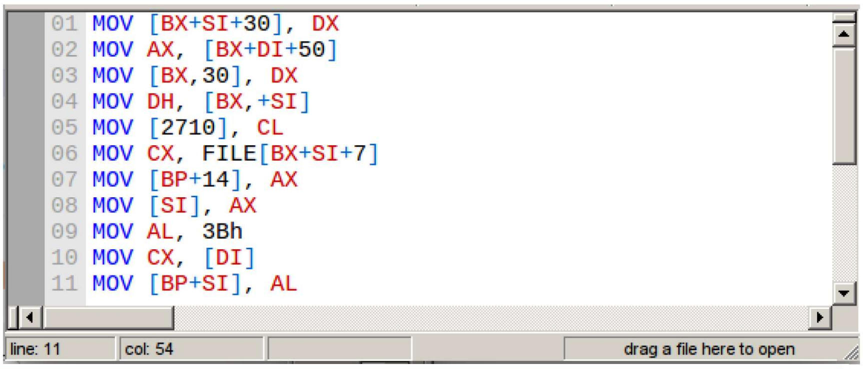
1. Write an assembly code to turn the green traffic light for east and west, and turn red light for north and south. Using:
   (a) Direct port addressing mode.
   (b) indirect port addressing mode.
2. Write an assembly code to turn on the first magnet of the stepper-motor. Using:
   (a) Direct port addressing mode.
   (b) indirect port addressing mode.

Note 6.1:

This material was acquired from the references (Page 77).

Questions of Chapter 6

3. Describable each line and write the equations of the physical address, also specify the type of addressing mode for the following:



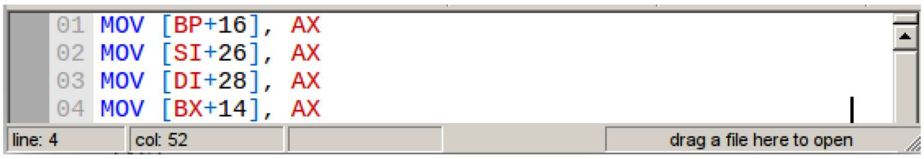
```
MOV [BX+SI+30], DX
MOV AX, [BX+DI+50]
MOV [BX,30], DX
MOV DH, [BX,+SI]
MOV [2710], CL
MOV CX, FILE[BX+SI+7]
MOV [BP+14], AX
MOV [SI], AX
MOV AL, 3Bh
MOV CX, [DI]
MOV [BP+SI], AL
```

Note 6.1:

This material was acquired from the references (Page 77).

Questions of Chapter 6

4. Calculate the physical address for the following assembly code, assume that DS = 4500 H, SS = 2000 H, BX = 2100 H, SI = 1486 H, DI = 8500 H, BP= 7814 H, and AX = 2512 H:



```
MOV [BP+16], AX
MOV [SI+26], AX
MOV [DI+28], AX
MOV [BX+14], AX
```

Note 6.1:

This material was acquired from the references (Page 77).