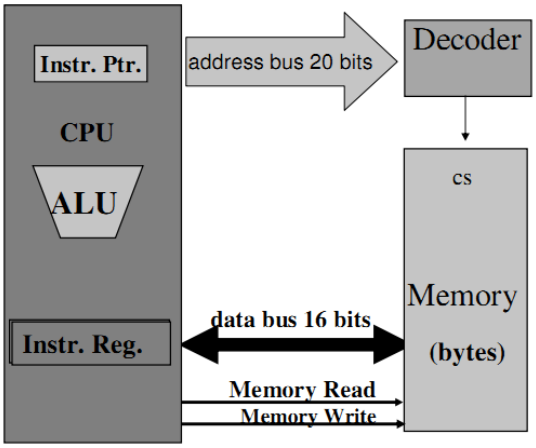
Three-bus Architecture

- ה- CPU מתזמן את תהליכי העברת מידע בין יחידת המחשב ומפעיל אותם: הזיכרון ורכיבי ה - I/O באמצעות פסי Address, Data ו- Control.
- התזמון נעשה באמצעות **מחזורי פס Bus Cycles**.
- ה - CPU מתחיל כל **מחזור הבאת הוראה** עם הבאת הוראה **Instruction Fetch** מיחידת הזיכרון. ה IP מקודם בהתאם לגודל ההוראה כהכנה למחזור Fetch הבא, וקוד הפעולה OPCODE מפוענח ומבוצע.

הבאת הוראה וביצועה
Fetch Execute Cycle



Fetch & Execute

- הבאת ההוראה מהזיכרון:
  כתובתה נמצאת תמיד ב - IP Register (או Instruction Pointer).
- הכנסת ההוראה לתוך אוגר ה – Instruction Register.
- קידום הכתובת באוגר ה-IP בהתאם לאורך ההוראה.
- פענוח הפקודה ב - Instruction Register וביצועה.
- חזרה לפעולה הראשונה.

מיעון ישיר Direct Addressing

- שיטת מיעון זו משמשת להגדרת אופרנד הנמצא בזיכרון . כתובת האופרנד בגוף ההוראה.
- אופרנד המוגדר בשיטה זו יכול להיות גם אופרנד המקור וגם אופרנד היעד .

מבנה :

MOV (dest) , [direct]

MOV [direct] , (source)

מיעון עקיף באמצעות אוגר
Indirect Addressing

- בשיטת מיעון זו הכתובת היחסית של האופרנד שמורה בתוך אוגר המשמש כמצביע .
- לאוגר המכיל את כתובת קוראים: אוגר ההצבעה Index או base.
- האוגרים היכולים לשמש כמצביע על נתון ב- Segment Data הם:

SI , DI , BX

האוגר היכול לשמש כמצביע על נתון בסגמנט המחסנית הוא :

כדי להשתמש בשיטת מיעון זאת יש לטעון את הכתובת של האופרנד לתוך אוגר ההצבעה , לפני מתן ההוראה.

בשיטה זו נוח להשתמש כאשר יש צורך לעבוד עם מערכים.

האופרנד המוגדר בשיטה זו יכול להיות אופרנד מקור או יעד .

מבנה :

MOV (dest) ,[pointing_register]

MOV [pointing_register] , (source)

```
AL = 11010011b   ; rolling left 1 position:
AL = 10100111b   ; rolling left 1 more position:
AL = 01001111b
```

```
rol x, y ; means : x is rolled left  by y positions
ror x, y ; means : x is rolled right by y positions
```

- *rcl* and *rcr*

```
CF = 0  AL = 11010011b ; rolling left 1 position
CF = 1  AL = 10100110b ; rolling left 1 more position
CF = 1  AL = 01001101b
```

מיעון עקיף - שילובים

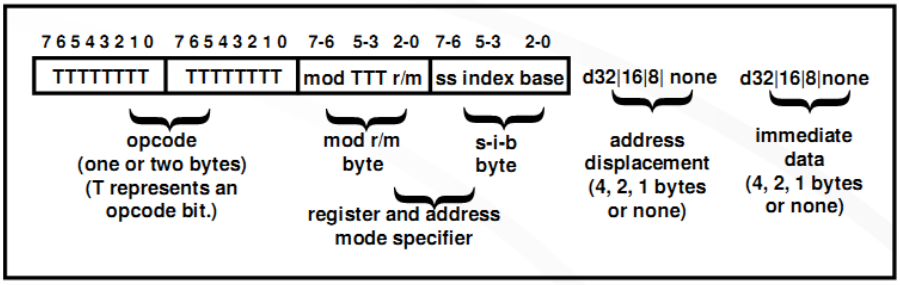
BASE	BX	MOV AL, [BX]
INDEX	SI, DI	MOV AL, [SI]
Basis&Index offset	BX SI / DI Offset	Mov al, [bx+si+30]

Instruction Format

INSTRUCTION PREFIX	ADDRESS-SIZE PREFIX	OPERAND-SIZE PREFIX	SEGMENT OVERRIDE	REP PREFIX	LOCK PREFIX
0 OR 1	0 OR 1	0 OR 1	0 OR 1	0 OR 1	0 OR 1
NUMBER OF BYTES					

OPCODE	MOD R/M	SIB	DISPLACMENT	IMMEDIATE
1 OR 2	0 OR 1	0 OR 1	0,1,2 OR 4	0,1,2 OR 4
NUMBER OF BYTES				

Figure 25-1. Instruction Format



Core Instruction Format

MACRO רוטינות ו

The main differences (behavior-wise) are:

- Macros use String replacement for its invocation whereas subroutines use Calls
- Due to replacement nature, macro can exist Multiple copies in the programs whereas subroutine can exist only in One copy
- Because of multiple copies possibility, you cannot obtain a macro's Address, whereas you can obtain a subroutine's address
- Macros can be faster since it doesn't have calling and return time penalty
- Macros can be harder to debug

קווי חומרה של המעבד
הקשורות לתהליך הפסיקה

כניסות:

1. NMI - Non Maskable interrupt - רגל פסיקה זו שימושית לתקלות מערכת קריטיות כגון נפילת מתח. במקרה כזה מעגל גילוי מזהה את נפילת המתח ושולח פסיקה המאפשרת למעבד לאחסן מידע רלוונטי מבעוד מועד.
2. INTR - הרגל מופעלת על ידי בקשת פסיקה. הרגל מיד עוברת לdisable כאשר הפסיקה מתקבלת על ידי המעבד ומתאפשרת שוב - (enable) כאשר נשלחת פקודת IRET או IRETD (ב386 ומעלה).

יציאות:

3. INTA - המעבד שולח פולס ברגל זו ומצפה לקבלת vector type number ב- data bus.

פירוט שלבי תהליך הפסיקה

1. התקן מבקש פסיקה מהבקר לדוגמה בקו IRQ3
2. בקר הפסיקות מעדכן את הבקשה ומקפיא את הבקשות האחרות באוגר IRR.
3. בקר הפסיקות שולח אות INT למעבד.
4. המעבד מחזיר אישור INTA.
5. הבקר מקבל את אישור הCPU ומעלה את סיבית המתאימה (3) באוגר ISR ל1 ואת הסיבית המתאימה (3) IRR מחזיר ל 0.
6. המעבד שולח INTA שני אשר גורם לבקר הפסיקות לשלוח את וקטור הפסיקה ב DATA BUS בהתאם למספר הקו IRQ
7. פסיקות מרמה עדיפות נמוכה או שווה נחסמות עד לשליחת פקודת Non specific EOI על ידי המתכנת בשיגרת ה- ISR. (=> איפוס סיבית מתאימה בISR).
8. סיום הפסיקה באמצעות IRET בISR – המעבד שולף את IP, CS, דגלים מהמחסנית.

❑ Fully Nested mode

- ❑ Default, אופן זה קובע את עדיפות המבואות IRQ לפי האינדקס:
  IRQ0 בעל העדיפות הגבוהה ביותר IRQ7 בעל העדיפות הנמוכה ביותר
  נכון לפסיקות המקבלות שרות – ולפסיקות המתקבלות בו- זמנית.
- ❑ בהתאם לערכו של הדגל IF וסוג ה- Non-Specific End of Interrupt EOI
  Automatic End of Interrupt

❑ Specially Fully Nested mode

מיועד לבקר הראשי במערכת הכוללת מספר בקרים מחוברים ב- cascade. פועל כמו Fully Nested Mode אך כולל הרחבה לגבי עדיפויות לבקרים המחוברים בטור.

❑ Non-Specific Rotating

מאפשר לתת עדיפות זהה לכל ההתקנים. כאשר נשלחת הוראת EOI באופן עבודה זה מאופסת סיבית ה-ISR המתאימה והקו המתאים מקבל את העדיפות הנמוכה ביותר. העדיפות של יתר המבואות מסובבות בהתאם.

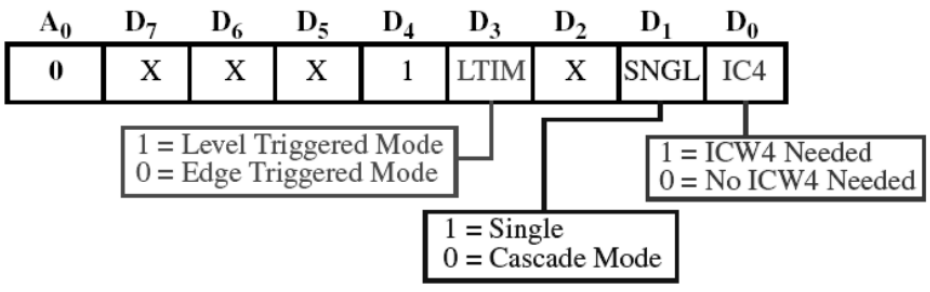
ניתן לתת את ההוראה של סיבוב ייחודי בזמן מתן שרות או באופן אוטומטי לאחר Inta . הוראת EOI מאפסת תמיד את סיבית הISR שמספרו הוא הנמוך ביותר.

באופן עבודה NonSpecific Rotating לא תמיד סיבית ה- ISR בעל המספר הנמוך ביותר היא הסיבית של קו הפסיקה המקבלת שרות.

- ❑ Specific Rotating אופן עבודה זה מאפשר סבב עדיפויות בין ההתקנים אבל EOI יכולה להצביע על סיבית ISR מסוימת שצריך לאפס כך שהקו שעבורו השרות הסתיים יקבל את העדיפות הנמוכה ביותר.

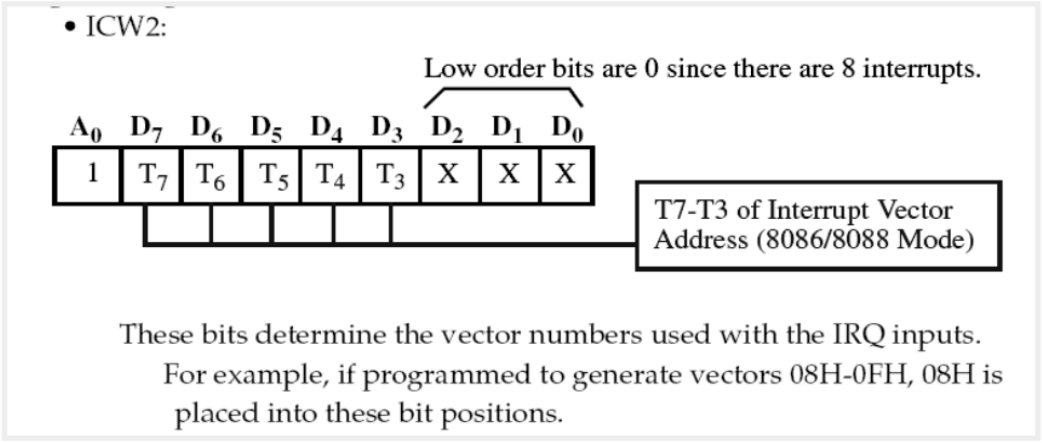
Initialization Command Word 1 (ICW1)

• ICW1:



Bit(s)	Function	
7:5 MSB	Interrupt Vector Addresses for MCS-80/85 Mode.	
4	Must be set to 1 for ICW1	
3	1	Level Triggered Interrupts
	0	Edge Triggered Interrupts
2	1	Call Address Interval of 4
	0	Call Address Interval of 8
1	1	Single PIC
	0	Cascaded PICs
0 LSB	1	Will be Sending ICW4
	0	Don't need ICW4

Initialization Command Word 2 (ICW2)

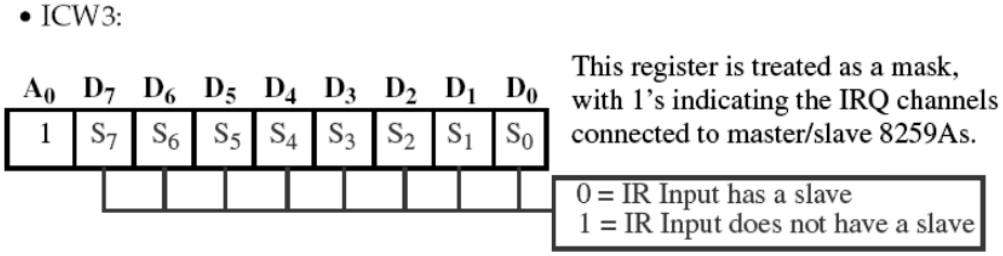


לשם מציאת הכתובת בזיכרון בה נמצאת הפיסקה אותה אתה כותב
נעשה shift 2 לערך הנתון ונקבל את המקון בו יושבת הפסיקה למשל shift 2 ל c8 ייתן לנו 320h

Initialization Command Word 3 (ICW3)

נדרש רק בחיבור

Cascade



A0=1

Initialization Command Word 3 for Master PIC (ICW3)

Bit	Function
7	IR7 is connected to a Slave
6	IR6 is connected to a Slave
5	IR5 is connected to a Slave
4	IR4 is connected to a Slave
3	IR3 is connected to a Slave
2	IR2 is connected to a Slave
1	IR1 is connected to a Slave
0	IR0 is connected to a Slave

Initialization Command Word 3 for Slaves (ICW3)

Bit(s)	Function	
7	Reserved. Set to 0	
6	Reserved. Set to 0	
5	Reserved. Set to 0	
4	Reserved. Set to 0	
3	Reserved. Set to 0	
2:0	*Slave ID*	
	000	Slave 0
	001	Slave 1
	010	Slave 2
	011	Slave 3
	100	Slave 4
	101	Slave 5
	110	Slave 6
	111	Slave 7

ICW4

Bit(s)	Function	
7	Reserved. Set to 0	
6	Reserved. Set to 0	
5	Reserved. Set to 0	
4	1	Special Fully Nested Mode
	0	Not Special Fully Nested Mode
3:2	0x	Non - Buffered Mode
	10	Buffered Mode - Slave
	11	Buffered Mode - Master
1	1	Auto EOI
	0	Normal EOI
0	1	8086/8080 Mode
	0	MCS-80/85

Operation Control Word 1 (OCW1)

Bit	PIC 2	PIC 1
7	Mask IRQ15	Mask IRQ7
6	Mask IRQ14	Mask IRQ6
5	Mask IRQ13	Mask IRQ5
4	Mask IRQ12	Mask IRQ4
3	Mask IRQ11	Mask IRQ3
2	Mask IRQ10	Mask IRQ2
1	Mask IRQ9	Mask IRQ1
0	Mask IRQ8	Mask IRQ0

Operation Control Word 2 (OCW2)

Bit(s)	Function	
7:5	000	Rotate in Auto EOI Mode (Clear)
	001	Non Specific EOI
	010	Reserved
	011	Specific EOI
	100	Rotate in Auto EOI Mode (Set)
	101	Rotate on Non-Specific EOI
	110	Set Priority Command (Use Bits 2:0)
	111	Rotate on Specific EOI (Use Bits 2:0)
4	Must be set to 0	
3	Must be set to 0	
2:0	000	Act on IRQ 0 or 8
	001	Act on IRQ 1 or 9
	010	Act on IRQ 2 or 10
	011	Act on IRQ 3 or 11
	100	Act on IRQ 4 or 12
	101	Act on IRQ 5 or 13
	110	Act on IRQ 6 or 14
	111	Act on IRQ 7 or 15

Operation Control Word 3 (OCW3)

Bit(s)	Function	
7	Must be set to 0	
6:5	00	Reserved
	01	Reserved
	10	Reset Special Mask
	11	Set Special Mask
4	Must be set to 0	
3	Must be set to 1	
2	1	Poll Command
	0	No Poll Command
1:0	00	Reserved
	01	Reserved
	10	Next Read Returns Interrupt Request Register
	11	Next Read Returns In-Service Register

הבחנה בין סוגי פסיקות

- **Hardware Interrupts**
old CS:EIP points past last completed instruction.
- **Traps** (INT 30, ...)
old CS:EIP points **past** instruction causing exception
- **Faults** (page fault, GPF, ...)
old CS:EIP points **to** instruction causing exception
- **Aborts** (hardware errors, bad system table vals...)
uncertain CS:EIP, serious problems, CPU confused

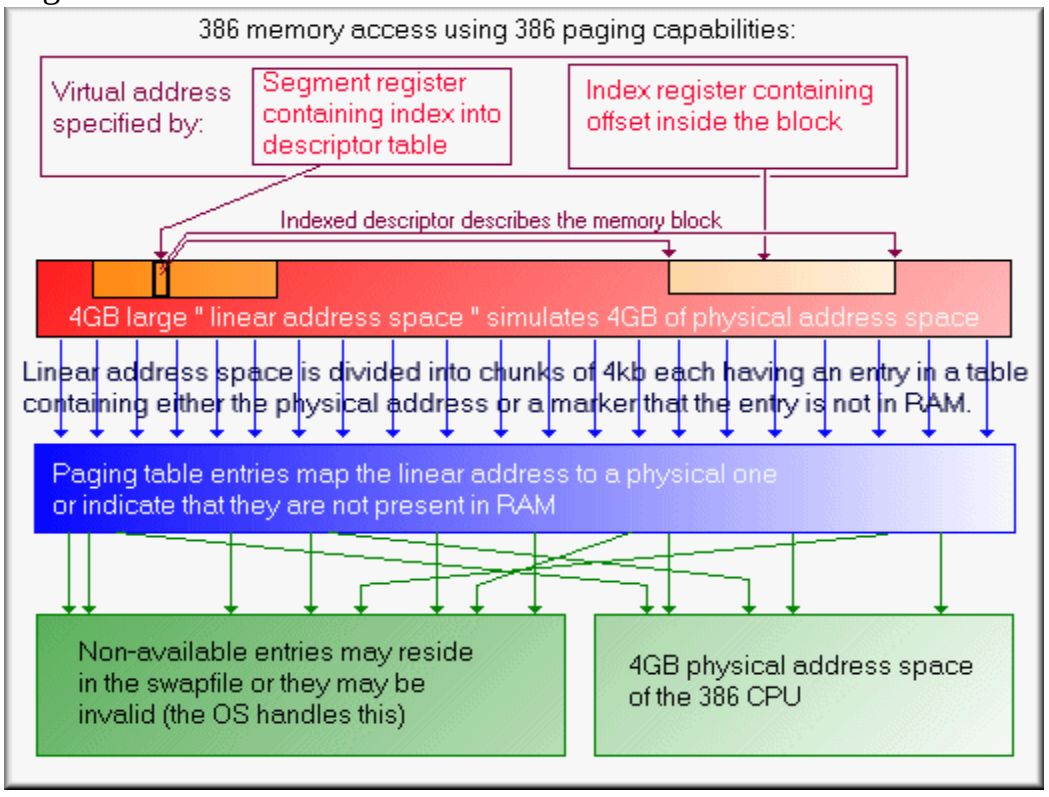
Exceptions

- Three types of exceptions
  - Depending on the way they are reported
  - Whether or not the interrupted instruction is restarted
    - Faults
    - Traps
    - Aborts
- Faults and traps are reported at instruction boundaries
- Aborts report severe errors
  - Hardware errors
  - Inconsistent values in system tables

Faults and Traps

- Faults
  - Instruction boundary before the instruction during which the exception was detected
  - Restarts the instruction
  - Divide error (detected during **div/idiv** instruction)
  - Segment-not-found fault
- Traps
  - Instruction boundary immediately after the instruction during which the exception was detected
  - No instruction restart
  - Overflow exception (interrupt 4) is a trap
  - User defined interrupts are also examples of traps

386 paging unit: This new processor extension operates completely separated from the normal Protected Mode memory addressing unit. The 4GB address space is divided into small chunks which can be moved around inside the RAM or they can be removed from RAM and stored onto the harddisk freeing RAM for other chunks.
Because it has nothing to do with the selector-offset addressing even programs creating or changing their own descriptors will not be able to recognize if paging is active or not. Memory access is now a three-layer thing:



Note: Of course, the memory described in the descriptor itself may be paged out there (see 286 part).

למה שיטת העימוד – PAGING עדיפה?

- בעיה של תהליך באורך משתנה
- בעיה של שבירה למקטעים Fragmentation של הזיכרון
- תהליך יכול להיות מחולק למקטעים לא מסודרים ברצף בזיכרון
- תהליך SWAP של תהליך דורש זמן ארוך
- עדיף לבצע SWAP למקטע של תהליך.

מצב המחסנית ב- Call & Call Near Far