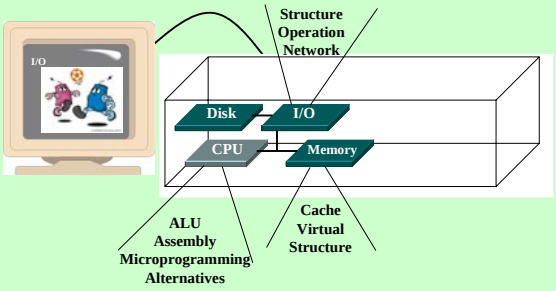
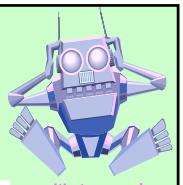
# נושאים של סמסטר ב'
## Semester Topics





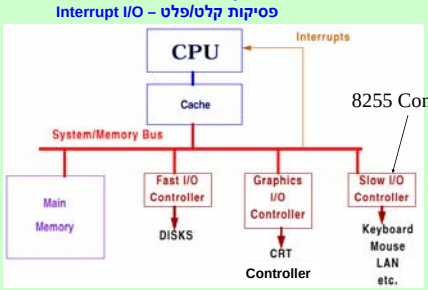


# Review – I/O - חזרות




- I/O Hardware **חומרה**

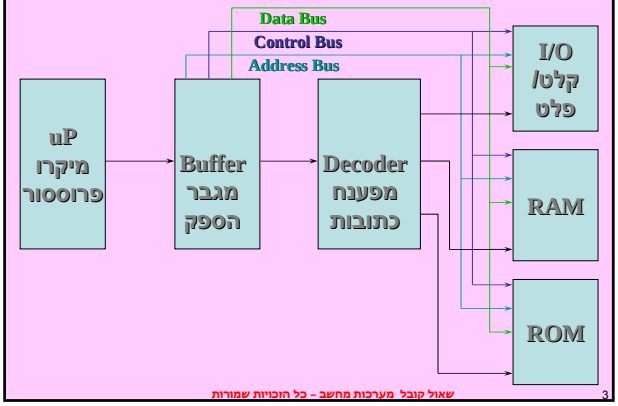
Memory Mapped I/O - מיפוי קלט/פלט
Programmed I/O – תכנות קלט/פלט
Interrupt I/O – פסיקות קלט/פלט




בקר קלט/פלט איטי



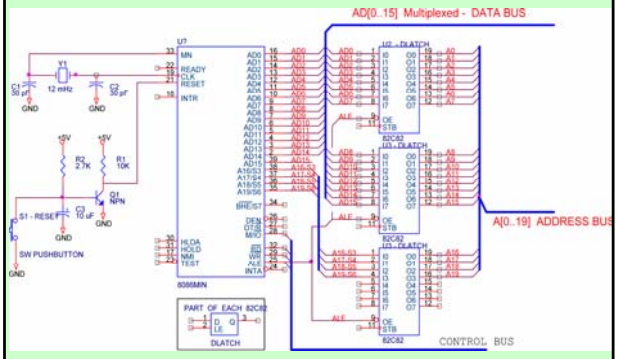

# חיבורי החומרה







# חיבורי המעבד 8086 הבסיסי

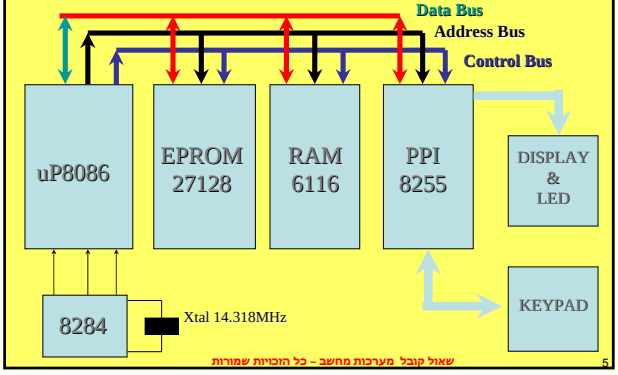






# דיאגראמה מלבנית מבוססת במעבד 8086

Block Diagram Micro Computer based on uP8086







# דוגמאות של מהירות התקני קלט / פלט

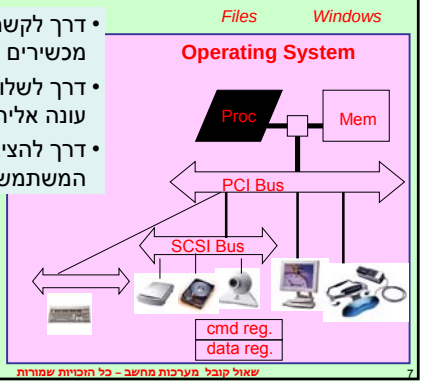
I/O Device Examples and Speeds

- I/O Speed: bytes transferred per second
(from mouse to display: million-to-1)

| • Device רכיב | Behavior קשר | Partner שותף | Data Rate (Kbytes/sec) |
|---|---|---|---|
| Keyboard | קלט | בן אדם | 0.01 |
| Mouse | קלט | בן אדם | 0.02 |
| Line Printer | פלט | בן אדם | 1.00 |
| Floppy disk | שמירה | מכונה | 50.00 |
| Laser Printer | פלט | בן אדם | 100.00 |
| Magnetic Disk | שמירה | מכונה | 10,000.00 |
| Network-LAN | קלט או פלט | מכונה | 100,000.00 |
| Graphics Display | פלט | בן אדם | 30,000.00 |

## מה אנחנו צריכים לעשות כדי להפעיל מכשירים של קלט/פלט?
## What do we need to make I/O devices work?

• דרך לקשר סוגים רבים של מכשירים לזיכרון של המעבד.
• דרך לשלוט במכשירים האלה, עונה אליהם, ומעביר נתונים.
• דרך להציג אותם לתוכניות המשתמש לכן הם שימושיים.






## I/O and Device Interfacing - שערי קלט/פלט

•שערים מיועדים להביא נתונים מהשער לתוך המעבד המרכזי (קלט) או לשלוח נתונים מהמעבד המרכזי לשער (פלט).
Ports are accessed either to bring data from the port into the CPU (in) or to send data from the CPU to the port (out).

•בנוסף לגישה לזיכרון, מיקרופרוססורים 80X86 הם מסוגלים לגשת לשערים.
In addition to memory, 80x86 microprocessors are able to access ports.

## Input/Output Instructions in the 8086 - פקודות קלט/פלט

•כאמור, מיקרופרוססור 8086 מסוגל לגשת לשערים בנוסף לגישה לזיכרון, וזה דרך הפקודות:
The 8086 microprocessor can access information from port as well as from memory.

• "OUT" (פלט) ו- "IN" (קלט)

•ההוראות האלה יכולות לשלוח נתונים מהצובר (AX, AL, or AH) לשערים או מקבל נתונים מהשערים לתוך הצובר.
These instructions can send data from the accumulator (AX, AL, or AH) to ports or receive data from ports into the accumulator.



## שערי קלט/פלט (סיכום)
Port input and output

- Used to communicate with many peripheral devices built into the processor, motherboard or expansion cards
- **IN** instruction transfers data from a device on the I/O bus to AL, AX, or EAX
- **OUT** instruction transfers data from AL, AX, or EAX to a device connected to the I/O bus



## 8086 - שערי קלט/פלט
Port input and output

Two forms of port addressing

- Fixed-port addressing
  - Allows an 8-bit I/O port address (use if port is 0 – 255)
  - port number is immediate (follows the instruction opcode)

```
IN   AL, 6Ah     ;data from I/O address 6Ah is input to AL
```

- Variable-port addressing
  - Allows using 16-bit port address (addresses up to FFFFh)
  - the I/O port number is stored in register DX

```
MOV DX, 0FA64h
OUT  DX, AX      ;transfers contents of AX to I/O port FA64h
```



**Case 1: 8-bit data ports**

| | Inputting Data | | Outputting Data | |
|---|---|---|---|---|
| Format | IN | dest,source | OUT | dest,source |
| (1) | IN | AL,port# | OUT | port#,AL |
| (2) | MOV | DX,port# | MOV | DX,port# |
| | IN | AL,DX | OUT | DX,AL |

• In format (1), port# is the address of the port and can be from 00 to FFH (256 ports), and no segment register is involved.

• In format (2), port# is the address of the port and can be from 0000 to FFFFH (65536 ports), and no segment register is involved.

The following code transfers the contents of register BL to port address 378H.

```
MOV   DX,378H     ;DX=378 the port address
MOV   AL,BL       ;load data into accumulator
OUT   DX,AL       ;write contents of AL to port whose address is in DX
```



**Case 2: 16-bit data ports**

| | Inputting Data | | Outputting Data | |
|---|---|---|---|---|
| (1) | IN | AX,port# | OUT | port#,AX |
| (2) | MOV | DX,port# | MOV | DX,port# |
| | IN | AX,DX | OUT | DX,AX |

• This requires two port addresses, one for each byte.

• Suppose AX = 98F6H and the port address is 47H

```
OUT     47H,AX     ;send out AX to port 47H & 48H
```

• F6H, the contents of AL, goes to port address 47H.

• 98H, the contents of AH, goes to port address 48H.

## 8086 I/O Bus Timing - תזמון בערוץ קלט/פלט

The concept of bus timing for I/O instruction is exactly the same as memory, with the following exceptions:

1. IOR and IOW are used instead of MEMR and MEMW. ( M/IO = 0 )
2. While the physical address for a memory location is always 20 bits, which are put on address bus A0-A19, the physical address for ports is 8 or 16 bits.
3. For an I/O port address, no segment register is used.



## Port I/O example - דוגמה

```
..START
    IN   al, 61h   ;read port 61h: PC speaker control
    register
    OR   al, 3     ;set bits 0 and 1
    OUT  61h, al   ;turn speaker on
    MOV  cx, 1000h ;delay count; if the count is increased,
                   ;the beep will become longer
.L1 LOOP .L1       ;time delay – spin 1000h times
    IN   al, 61h
    AND  al, 0fch  ;clear bits 0 and 1
    OUT  61h, al   ;turn speaker off

    MOV  ax, 4c00h ;Normal DOS Exit
    INT  21h
```

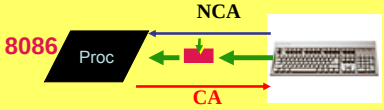


## Keyboard Interface: Handshake

**לחיצה בכפתור המקלדת גורם לה לייצר אות בקרה:**
**A key pressed is indicated by the keyboard by Signal:**
***New Character Available* (NCA)**

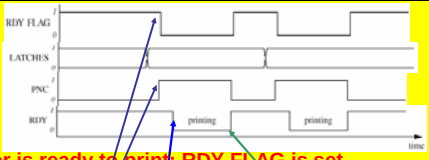
***תאי זיכרון שומרות נתוני הלחיצות של המקלדת***
***Data Latches* store the key data when key pressed**

***אות מהיע"מ מודיע שכבר קרא נתון המקלדת***
***Character Accepted* (CA)**
**signal from CPU indicates CPU has read the key data**



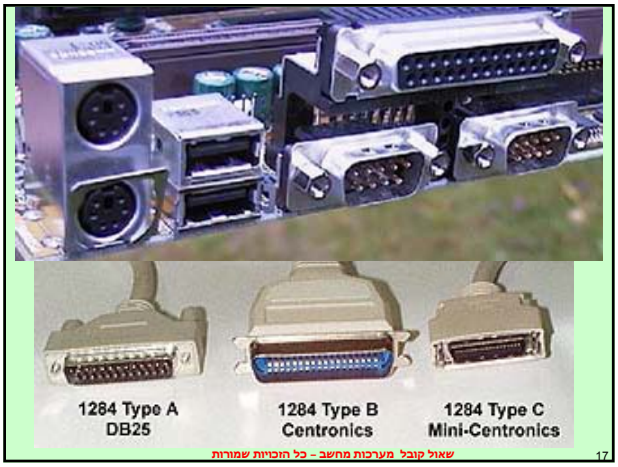



## Printer Interface: Sequence of Operations




1. Printer is ready to print. RDY FLAG is set.
2. CPU can examine the RDY FLAG at any time by reading from the RDY FLAG (address $1003).
3. If it finds RDY FLAG set, it can write the ASCII Data for the character to the LATCHES (address = $1002)
4. CPU clears the RDY FLAG . Inverse of RDY FLAG is the PNC signal for the printer
5. The printer will then respond to PNC by reading the LATCHES, negating the RDY signal, and starting to print.
6. When printing finishes, printer re-asserts RDY (which sets RDY FLAG).

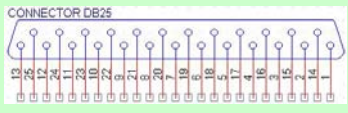






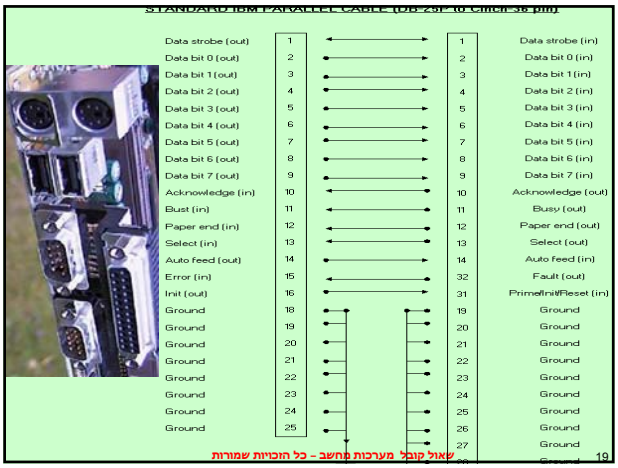

## Printer Interface - החיבור המקבילי

החיבור המקבילי של המחשב מספק לנו גישה של 17 הדקים אשר מאפשרים לנו מעבר מהיר של מידע בין השקע למחשב. אנו מכירים את החיבור של המדפסת למחשב (לפחות בדגמים הישנים של המדפסת כי היום יש מעבר לחיבור מדפסת דרך מיפתח ה- USB).
המשמעות של מידע מקבילי הוא בכך שניתן להעביר דרך החיבור Byte אחד (כלומר 8 ביט של מידע) בכל פעם. להבדיל מהחיבור הטורי שמאפשר להעביר ביט אחד של מידע בכל פעם.
ן לכל הדק תפקיד ושם משלו. אנו נתייחס לשמות ההדקים בשמם המקורי ולפי הצורך **נגדיר לכל הדק תפקיד משלו** בהתאם לצרכים שלנו.
**המבנה הפיזי של החיבור.**
המבנה הפיזי של החיבור הוא מחבר מסוג D-type בן 25 הדקים



| כיוון | מספר החדק | צבע חוט | שם האות | סיבית באוגר | תפקיד |
|---|---|---|---|---|---|
| יציאה | 1 | שחור | Strobe | C0 | Set Low p1ulse >0.5 us to send |
| יציאה | 2 | חום | Data 0 | D0 | ביט0 - LSB |
| יציאה | 3 | אדום | Data 1 | D1 | ביט1 |
| יציאה | 4 | כתום | Data 2 | D2 | ביט2 |
| יציאה | 5 | צהוב | Data 3 | D3 | ביט3 |
| יציאה | 6 | ירוק | Data 4 | D4 | ביט4 |
| יציאה | 7 | כחול | Data 5 | D5 | ביט5 |
| יציאה | 8 | סגול | Data 6 | D6 | ביט6 |
| יציאה | 9 | אפור | Data 7 | D7 | ביט7 -MSB |
| כניסה | 10 | לבן | Ack | S6 | IRQ; Low Pulse ~ 5 uS, after accept |
| כניסה | 11 | ורוד | Busy | S7 | High for Busy/Offline/Error |
| כניסה | 12 | ירוק בהיר | Paper End | S5 | High for out of paper |
| כניסה | 13 | שחור לבן | Select In | S4 | High for printer selected |
| יציאה | 14 | חום לבן | Auto Fd | C1 | Set Low to auto feed one line |
| כניסה | 15 | אדום לבן | Error | S3 | Low for Error/Offline/Paper End |
| יציאה | 16 | כתום לבן | Init | C2 | Set Low pulse > 50uS to init |
| יציאה | 17 | ירוק לבן | Select | C3 | Set Low to select printer |
| אין | 18-25 | כחול לבן | Ground | | |

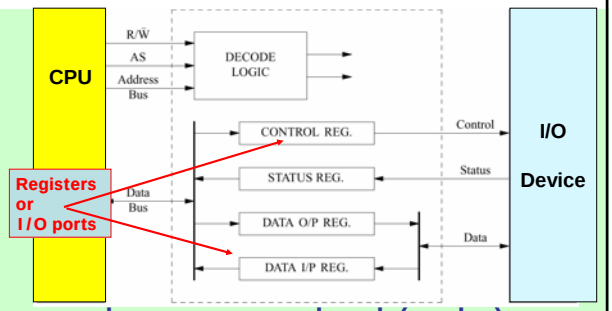







# בקרים סטנדרטיים של קלט/פלט

כדי להקל לממשק של ערוץ עם מכשירים חיצוניים פשוטים




**המודים (modes) להפעלה וקונפיגורציות של שערים, לעיתים קרובות ניתנות לתכנות.**



**רמות מתח:**

תואמות את רמות המתח המקובלות בתקן TTL ואלה הם:

| מוצא | | כניסה | |
|---|---|---|---|
| הרמה הלוגית | המתח המוצא | מתח בכניסה | הרמה הלוגית |
| 1 לוגי | 5V | נתק או 5V | 1 לוגי |
| 0 לוגי | 0V | אדמה 0V | 0 לוגי |

**התנגדות הכניסה:**

התנגדות של 4.7K אום.

**זרם מוצא מקסימאלי:**

זרמי המוצא משתנים מאד ממחשב למחשב. אפשר לסמוך לפחות על 2mAבמצב של 1 לוגי ועל 12mA במצב של 0 לוגי

פורט המוצא מחובר עם נגדים של 4.7K אום אל המתח החיובי לכן, יכולת דחיפת הזרם של יציאה מסוג זה היא מוגבלת מאד.

מניסיון שלי לא ניתן להתבסס על מתחים אלה להפעלת רכיבים חיצוניים אפילו לא לידם בגלל הזרם הנמוך שמספק החיבור על כן בכל במעגלים שיוצגו בהמשך אני מסתפק בהדקים כהדקי אותות, ולרכיבים מסופק מתח חיצוני. (חשוב במצב זה לשתף את האדמה של החיבור לאדמה של מקור המתח החיצוני). אפשרות אחרת, אם אין מקור מתח חיצוני אחר, היא לשתף במעגל את שקע המקלדת. בשקע זה מופיע מתח של 5Vואפשר לנצלו למעגל החיצוני.

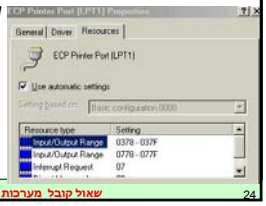


# Parallel interface - ממשק מקבילי

- כל התקן לוגי מקבל כתובת יחיד של שער קלט/פלט

| Device | Data Port | Status Port | Control Port |
|---|---|---|---|
| LPT0: | 3BCh | 3BDh | 3BEh |
| LPT1: | 378h | 379h | 37Ah |
| LPT2: | 278h | 279h | 27Ah |

- The parallel port base addresses are stored at 40:0008h, and can be viewed by using *debug* or *td*
  - At the command prompt, type "debug" to get the "-" debug prompt
  - type **d40:0008** (and hit return)
  - you get : **0040:0008**
    **BC 03 78 03 78 02** …
  - the first six bytes are addr of the installed printer adapters






# Parallel interface - ממשק מקבילי

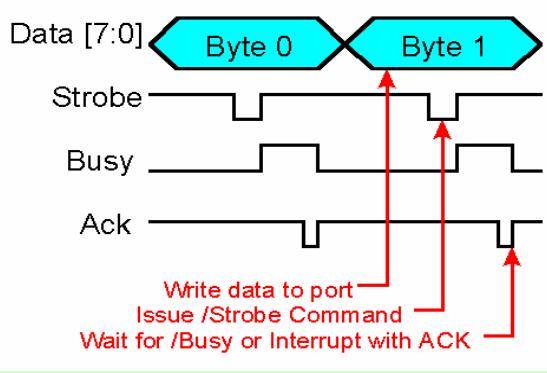
- בתחילה עוצב לשימוש עם מדפסות בלבד.
- מספק 8 סיביות להעברה של נתונים (בתחילה רק פלט, עכשיו דו כיווני).
- מספק חיבורי קלט/פלט ברמה של TTL (5V).
- בזמן אתחול המחשב, זה מגלה כל השערים מקביליים וקובע 'שערים לוגיים', וממיר את שמותיהם.(LPT1, LPT2, LPT3)
- כל שער מקבילי משתמש בשלוש שערים I/O של המעבד:
  - יציאת נתונים – data output,כדי לשלוח בית (byte) של נתונים למדפסת, שולחים הנתון לאוגר יציאה (output data register)
  - אותות מצב – status מהמדפסת למחשב
  - פקודות / בקרה – control/commands מהמחשב למדפסת

## אותות עקרים של הממשק
## Parallel interface major signals

| Signal(s) | Input/Output | Notes |
|---|---|---|
| Data[7:0] | Bi-directional | Byte-wide data bus |
| Strobe | To device | Write signal |
| Busy | From device | Don't send more data |
| Ack | From device | Acknowledge interrupts |
| Initialize | To device | Initialize external device |
| Out of Paper | From device | Status signal |



## כתיבה לממשק המקביל
## Writing to the parallel interface






## שערי בקרה ומצב - Control & Status Ports

| Control Port | |
|---|---|
| Bit | Description |
| 7-5 | Unused |
| 4 | IRQ Enable |
| 3 | Select [inverted] |
| 2 | Initialize |
| 1 | AutoFeed [inverted] |
| 0 | Strobe [inverted] |

| Status Port | |
|---|---|
| Bit | Description |
| 7 | Busy [inverted] |
| 6 | Ack |
| 5 | Out-of-Paper |
| 4 | Selected |
| 3 | I/O Error |
| 2-0 | Unused |



## PC Parallel Port

1981 - IBM PC printer port - transfer 8-bits in parallel to printer

Problems

1. PC performance has increased substantially so original parallel port is not adequate. Max. speed ~ 150KByte/s.
2. Originally only for output, no standard electrical interface.
3. Lack of standards limited cable length.

1991 - Discussion of a new standard began. Want a high speed bidirectional parallel port for the PC which is fully compatible with the original parallel port software and peripherals, but with a data rate greater than 1 megabyte/second in and out.



## The IEEE 1284 Standard (1994)

***Standard Signaling Method for a Bi-directional Parallel Peripheral Interface for Personal Computers***

Overview
- 4 control lines
- 5 status lines
- 8 data lines (originally out only, later bi-directional)

Ports (original PC)
Base address 278H, 378H, or 3BCH

There are three ports for each base address,
e.g., 378H, 379H, 37AH.
The last port was on the monochrome monitor card of the original PC.



The 1284 standard has 8 to 16 ports starting at 378H or 278H or can relocate the ports.

**IEEE 1284 Data Transfer Modes**
Can have forward and reverse channel connections, only one set of data lines, therefore half duplex.

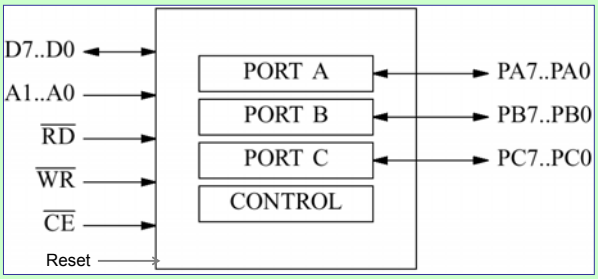
Compatibility and nibble modes - any existing parallel port (input is 4-bits using status lines).
Compatibility and byte modes - need direction bit in the control register. Can now input on external data lines.

EPP - Enhanced Parallel Port. CD-ROM, zip, tape drives
ECP - Extended Capability Port. Scanners, new printers
EPP and ECP require hardware to implement a state machine capable
of automatically generating control strobes needed for high performance data transfer modes.

## מתאם המקביל: Intel 8255 PIA

ארבעה רגיסטרים עם-גישה לקונפיגורציה של I/O ל-PIA

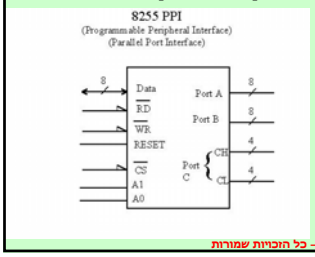
Four Addressable registers for PIA configuration and I/O operations

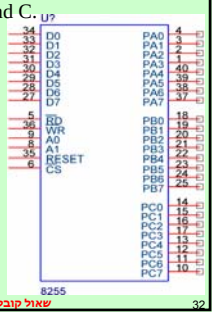





## 8255 Programmable Peripheral Interfacing Chip

- It can be used for both memory mapped and peripheral I/O.
- One can program the individual ports of the 8255 to be input or output, and change them dynamically.
- It has three separately accessible ports, A, B, and C.
- The 8255 is a 40-pin DIP chip.






**PA0-PA7**
This 8-bit port A can be programmed all as input or all as output or all bits as bidirectional input/output.
**PB0-PB7**
This 8-bit port B can be programmed all as input or all as output. Port cannot be used as a bidirectional input/output.

**PC0-PC7**

This 8-bit port C can be programmed all as input or all as output. It can also split into two parts, CU and CL. Each can be used for input or output. In addition, any of PC0-PC3 can be programmed.

**RD and WR** Control from cpu , read or write to ppi

**RESET**

This is an active-High signal input into the 8255 used to clear the control register. When RESET is activated, all ports are initialized as input ports.

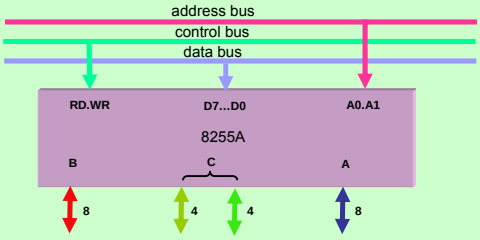
**A0, A1 and CS**

To select port A, port B, port C, and control register. (refer to Table 4-1)



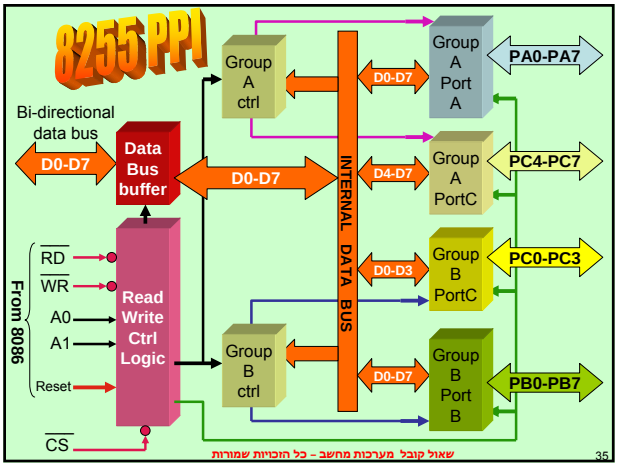
## 8255 External Description

Microcomputer bus general purpose I/O component

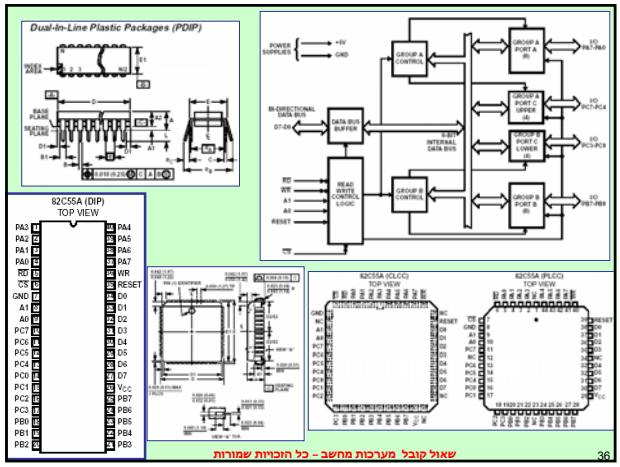



11/09/'98







## 8255 Programmable Peripheral Interfacing Chip

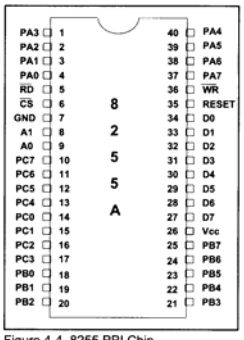



Figure 4-4. 8255 PPI Chip
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)

### גישה לחלק הפנימי של 8255

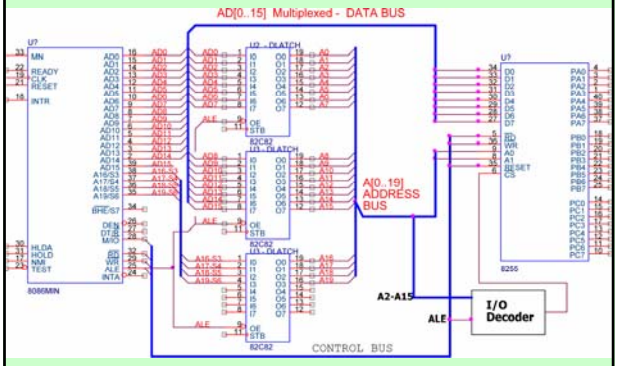
Table 4-1: 8255 Port Selection

| CS* | A1 | A0 | Selects: |
|---|---|---|---|
| 0 | 0 | 0 | Port A |
| 0 | 0 | 1 | Port B |
| 0 | 1 | 0 | Port C |
| 0 | 1 | 1 | Control register |
| 1 | x | x | 8255 is not selected |

(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)




## חיבורי מיקרו לרכיב 8255







## גישה לחלק הפנימי של 8255

### Operation Word Format

| A1 | A0 | RD | WR | CS | input operation ( read ) |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | port A - data bus |
| 0 | 1 | 0 | 1 | 0 | port B - data bus |
| 1 | 0 | 0 | 1 | 0 | port C - data bus |
| 1 | 1 | 0 | 1 | 0 | control word - data bus |
| A1 | A0 | RD | WR | CS | output operation ( write ) |
| 0 | 0 | 1 | 0 | 0 | data bus - port A |
| 0 | 1 | 1 | 0 | 0 | data bus - port B |
| 1 | 0 | 1 | 0 | 0 | data bus - port C |
| 1 | 1 | 1 | 0 | 0 | data bus - control |
| | | | | | Disable function |
| x | x | x | x | 1 | data bus - 3 - state |
| x | x | 1 | 1 | 0 | data bus - 3 - state |

11/09/98




**Mode Selection of the 8255A**

While ports A, B, and C are used for I/O data, it is the control register that must be programmed to select the operation mode of the three ports A, B, and C.

1. Mode 0, simple I/O mode. In this mode, any of the ports A, B, CL, and CU can be programmed as input or output. In this mode, all bits are out or all are in.
2. Mode 1. In this mode, ports A and B can be used as input or output ports with handshaking capabilities. Handshaking signals are provided by the bits of port C.
3. Mode 2. In this mode, Port A can be used as a bidirectional I/O port with handshaking capability whose signal s are provided by port C. Port B can be used either in simple I/O mode or handshaking mode 1.
4. BSR (bit set/reset) mode. In this mode, only the individual bits of port C can be programmed.



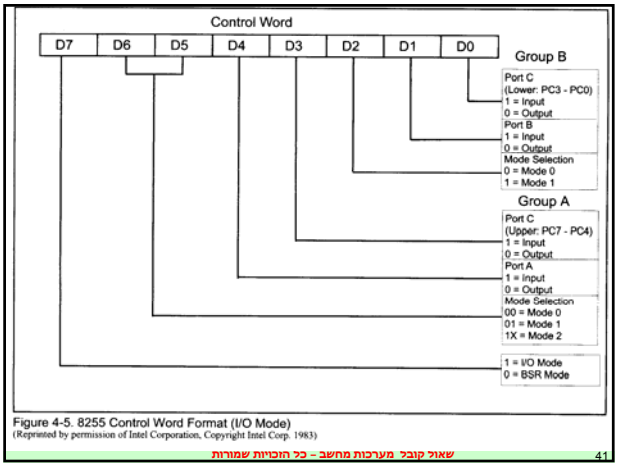
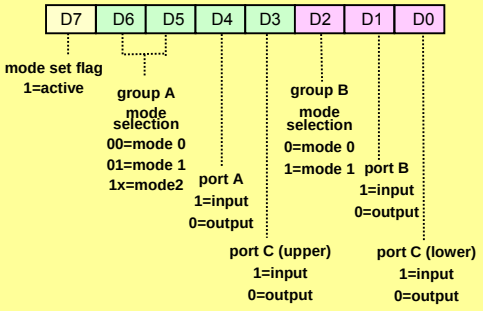




Figure 4-5. 8255 Control Word Format (I/O Mode)
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)




## דרך תכנות אוגר הבקרה
### Operation Definition Format

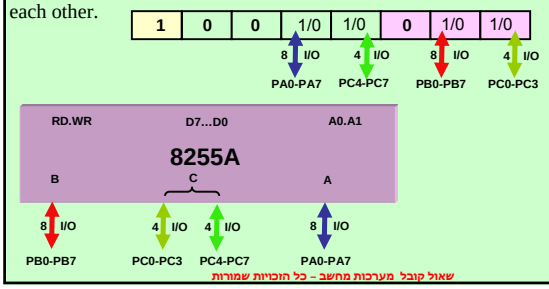


## Mode 0: Simple Input or Output

• Intel calls mode 0 the basic input output mode (or simple I/O).

• In this mode a given port cannot be both an input and output port at the same time.

• Port can program CL (PC0-PC3) and CU (PC4-PC7) independent of each other.






**Example 4-5**

The 8255 shown in Figure 4-6 is configured as follows: port A as input, B as output, and all the bits of port C as output.
(a) Find the port addresses assigned to A, B, C, and the control register.
(b) Find the control byte (word) for this configuration.
(c) Program the ports to input data from port A and send it to both ports B and C.

**Solution:**

(a) The port addresses are as follows:

| CS* | A1 | A0 | Address | Port |
|---|---|---|---|---|
| 0101 00 | 0 | 0 | 50H | Port A |
| 0101 00 | 0 | 1 | 51H | Port B |
| 0101 00 | 1 | 0 | 52H | Port C |
| 0101 00 | 1 | 1 | 53H | Control register |

(b) The control word is 90H, or 1001 0000.

(c) One version of the program is as follows:

```
MOV   AL,90H      ;control byte PA=in, PB=out, PC=out
OUT   53H,AL      ;send it to control register
IN    AL,50H      ;get the data from PA
OUT   51H,AL      ;send it to both PB
OUT   52H,AL      ; and PC
```



Using the EQU directive one can rewrite the above program as follows:

```
PORTA   EQU   50H
PORTB   EQU   51H
PORTC   EQU   52H
CNTLREG EQU   53H
...
        MOV   AL,90H          ;control byte PA=in, PB=out, PC=out
        OUT   CNTLREG,AL      ;send it to control register
        IN    AL,PORTA        ;get the data from PA
        OUT   PORTB,AL        ;send it to both PB
        OUT   PORTC,AL        ; and PC
```

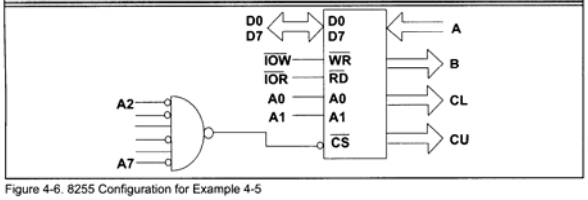



Figure 4-6. 8255 Configuration for Example 4-5



**Example 4-6**

(a) Find the port address for Figure 4-7.
(b) Find the control word if PA =out, PB=in, PC0 - PC3 =in, and PC4 - PC7=out.
(c) Program the 8255 to get data from port A and send it to port B. In addition, data from PCL is send out to the PCU.

**Solution:**

(a) The port addresses are as follows:

| CS* | A1 | A0 | Address | Port |
|---|---|---|---|---|
| 0111 11 | 0 | 0 | 7CH | Port A |
| 0111 11 | 0 | 1 | 7DH | Port B |
| 0111 11 | 1 | 0 | 7EH | Port C |
| 0111 11 | 1 | 1 | 7FH | Control register |

(b) The control word is 83H, or 1000 0011.

(c) The code is as follows.

```
MOV   AL,83H      ;control byte PA=out, PB=in, PCL=in, PCU=out
OUT   7FH,AL      ;send it to control register
IN    AL,7DH      ;get the data from PB
OUT   7CH,AL      ;send it to PA
IN    AL,7EH      ;get the bits from PCL
AND   AL,0FH      ;mask the upper bits
ROL   AL,1
ROL   AL,1        ;shift the bits
ROL   AL,1        ;to upper position
ROL   AL,1
OUT   7EH,AL      ;send it to PCU
```



Alternately, the four instructions above of "ROL AL,1" could be replaced with the following two instructions:

```
MOV   CL,4        ;count = 4
ROL   AL,CL       ;rotate 4 times
```

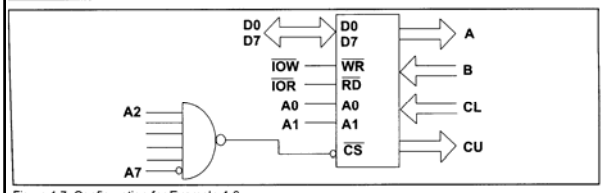



Figure 4-7. Configuration for Example 4-6

**BSR (Bit/Set/Reset) Mode**

BSR mode allows one to set to high or low any of PC0-PC7 as shown in Fig. 4-8.



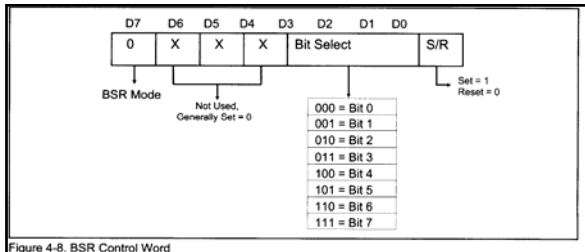



Figure 4-8. BSR Control Word
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)

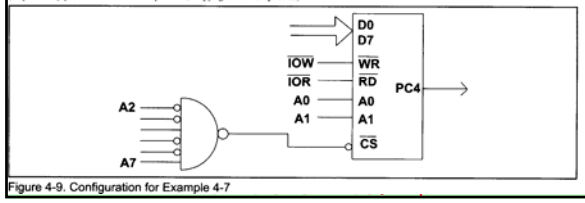



Figure 4-9. Configuration for Example 4-7

**Example 4-7**

Program PC4 of the 8255 in Figure 4-9 to generate a pulse of 50 ms with 50% duty cycle.

**Solution:**
To program the 8255 in BSR mode, bit D7 of the control word must be low. For PC4 to be high, we need a control word of "0xxx1001". Likewise, for low we would need "0xxx1000" as the control word. The x's are for "don't care" and generally are set to zero.

```
MOV   AL,00001001B    ;load the control byte (PC4=1)
OUT   93H,AL          ;set PC4 to high, sent to control reg
CALL  DELAY           ;time for the high part of pulse
MOV   AL,00001000B    ;load the control byte (PC4=0)
OUT   93,AL           ;set PC4 to low, sent to control reg
CALL  DELAY           ;time for the low part of pulse
```

In the above program, in the instruction "MOV AL,00001001B" the B stands for binary. There are various methods of writing a DELAY subroutine. Some are shown in Chapter 5.



**Example 4-8**

Program the 8255 in Figure 4-9 for the following.
(a) Set PC2 to high.
(b) Use PC6 to generate a square wave of 66% duty cycle continuously.

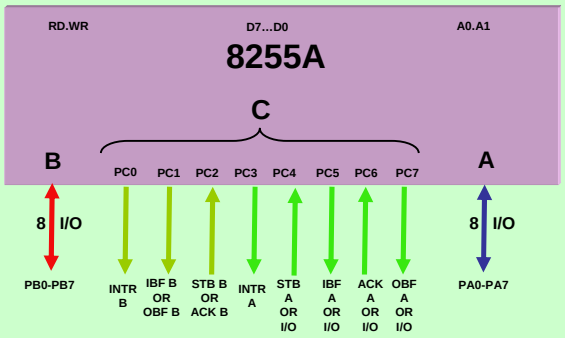
**Solution:**

(a)
```
        MOV   AL,0xxx0101     ;LOAD THE CONTROL BYTE (PC2=1)
        OUT   93H,AL          ;SET PC2=HIGH
```

(b)
```
AGAIN:  MOV   AL,0xxx1101     ;load the control byte (PC6=1)
        OUT   93H,AL          ;set PC6 to high, sent to control reg
        CALL  DELAY           ;time for the high part of pulse
        CALL  DELAY           ;one more time
        MOV   AL,0xxx1100     ;load the control byte (PC6=0)
        OUT   93,AL           ;set PC6 to low, sent to control reg
        CALL  DELAY           ;time for the low part of pulse
        JMP   AGAIN           ;stay in loop indefinitely
```



# MODE 1






**8255 in Mode 1: I/O with Handshaking Capability**

- One of the most powerful features of the 8255 is its ability to handle handshaking signals.
- Handshaking refers to the process of communicating back and forth between two intelligent devices.

Let us take the printer as an example:

1. A byte of data is presented to the data bus of the printer.
2. The printer is informed of the presence of a byte of data to be printed by activating its STROBE input signal.
3. Whenever the printer receives the data it informs the sender by activating an output signal called ACK (acknowledge)
4. The ACK signal initiates the process of providing another byte of data to the printer.



Mode 1: Outputting Data with Handshaking Signals

• As shown in Fig. 4-10, A and B can be used as output ports to send data to a device with handshaking signals.

• The handshaking signals for both ports A and B are provided by the bits of Port C.

• Fig. 4-11 provides a timing diagram.

**$\overline{\text{OBF}}$a** (output buffer full for port A)

• This is an active-low signal going out of PC7 to indicate the fact that the CPU has written a byte of data into port A.

• $\overline{\text{OBF}}$a must be connected to STROBE of the receiving equipment (such as a printer) to inform it that now it can read a byte of data from the port A latch.



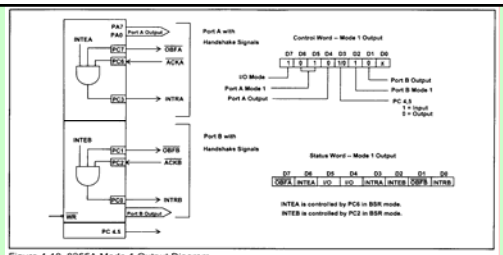


Figure 4-10. 8255A Mode 1 Output Diagram
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)

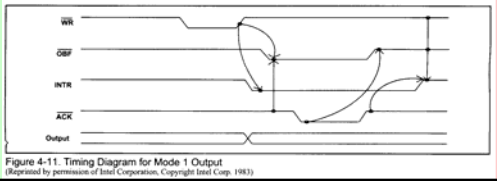



Figure 4-11. Timing Diagram for Mode 1 Output
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)

**$\overline{\text{ACK}}$a (acknowledge for port A)**

• This is an active low input signal received at PC6 of the 8255.

• Through $\overline{\text{ACK}}$, the 8255 knows that the data at port A has been picked up by the receiving device.

• The 8255 in turn makes $\overline{\text{OBF}}$a high, to indicate that the data at the port is old data now.

• $\overline{\text{OBF}}$a will not go low until the CPU writes a new byte of data to PORT A.

**INTRa (interrupt request for port A)**

•This is an active-high signal coming out of PC3.

• The $\overline{\text{ACK}}$ signal is a signal of limited duration. When it goes active (low) it makes $\overline{\text{OBF}}$a inactive, stays low for a small amount of time and then goes back to high (inactive)

• It is the rising edge of $\overline{\text{ACK}}$ that activates INTRa by making it high.

• This high signal on INTRa can be used to get the attention of the CPU.



**INTEa (interrupt enable for port A)**

• To disable INTRa to prevent it from interrupting the CPU.

• INTEa is controlled by PC2 in BSR mode.

**Status word**

• The 8255 enables monitoring the status of signals INTR, OBF, and INTE for both ports A and B.

• This is done by reading port C into the accumulator and testing the bits.

**Interrupts vs. Polling**

• There are two ways for the CPU to provide service to those devices: interrupt and polling

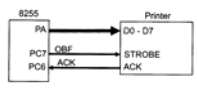
• In the interrupt method, whenever any device needs its service, the device informs the CPU by sending it an interrupt signal.

• In polling, the CPU continuously monitors a status condition and when the conditions are met it will perform the service.



**Example 4-9**

The diagram below shows the connection between the 8255 and a printer. Write a program to print the following messages: "Hi. How are you?", "I am fine. How are you?","$". The $ sign indicates the end of the message.




**Solution:**

From the data segment:

```
MY_DATA   DB    "Hi. How are you?",CR,LF
          DB    "I am fine. How are you?",CR,LF,"$"
PA        EQU   300H        ;port A address
PB        EQU   301H        ;port B address
PC        EQU   302H        ;port C address
CWP       EQU   303H        ;control word port
LF        EQU   0AH         ;line feed
CR        EQU   0DH         ;carriage return
```

From the code segment:

```
          MOV   AL,10100000B    ;control word PA=out mode 1
          MOV   DX,CWP          ;DX =  303 control word  port
          OUT   DX,AL           ;issue control word
          MOV   AL,00001101B    ;PC6=1 for INTEa
          OUT   DX,AL           ;using BSR mode
          MOV   SI,OFFSET MY_DATA ;SI = data address
AGAIN:    MOV   AH,[SI]         ;get a character
          CMP   AH,'$'          ;is it the end?
          JZ    OVER            ;if yes, exit
          MOV   DX,PC           ;DX=302  port C address
BACK:     IN    AL,DX           ;get status byte from  port C
          AND   AL,08           ;is INTRa high?
          JZ    BACK            ;if no, keep checking
          MOV   DX,PA           ; if yes, make DX=300 data port
          MOV   AL,AH           ;address and
          OUT   DX,AL           ;send char to printer
          INC   SI              ;increment the data  pointer
          JMP   AGAIN           ;keep doing it
OVER:     ...                   ;go back to DOS
```



**Mode 1: input ports with handshaking signals**

The 8255 can be programmed to receive data through ports A and B using handshaking signals through port C.

**$\overline{\text{STB}}$ (strobe)**

• This is an active-low input signal.

• When an external peripheral device provides a byte of data to an input port (A or B), it informs the 8255 through the $\overline{\text{STB}}$ pin that it can load (latch in) the data into its internal register.

**IBF (input buffer full)**

• This is an active-high output signal.

• In response to $\overline{\text{STB}}$, the 8255 latches into its internal register the data present at PA0-PA7 or PB0-PB7, and through IBF indicates that it has latched the data, but has not been read by the CPU yet.



**INTR (interrupt request)**

• This is an active-high output signal.

• If INTE = 1 when IBF goes active, INTR is activated (set to high) to inform the CPU that there is a byte of data in the 8255.

• The falling edge of $\overline{\text{RD}}$ makes INTR go low.

**INTE (interrupt enable)**

• An internal flip-flop can be used to enable or disable (mask) INTR generation.

• It is controlled by PC4 and PC2 in BSR mode.

• To control INTEa and INTEb, use PC4 and PC2, respectively.

**Status word**

• To allow implementation of polling, the status of the handshaking signals provided by port C can be checked by reading port C.



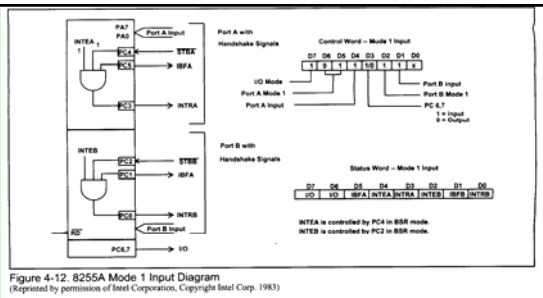



Figure 4-12. 8255A Mode 1 Input Diagram
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)

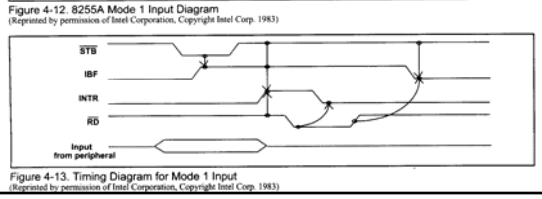
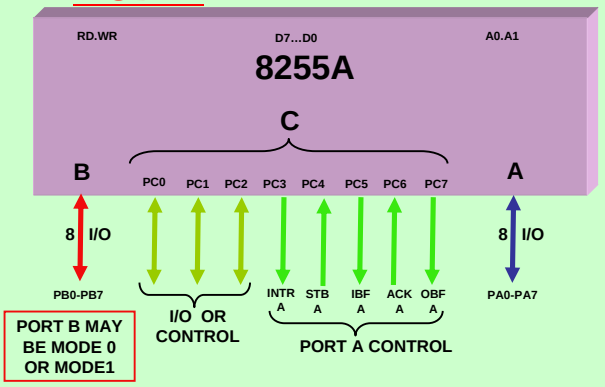



Figure 4-13. Timing Diagram for Mode 1 Input
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)

## MODE 2






## Mode 2: Bidirectional I/O with handshaking

• To transfer data both in and out via the same port with handshaking capability, port A is used in Mode 2.

• The handshaking signals for port A are provided by selected bits of port C.

• We can also configure Port B in either mode 0 or 1.

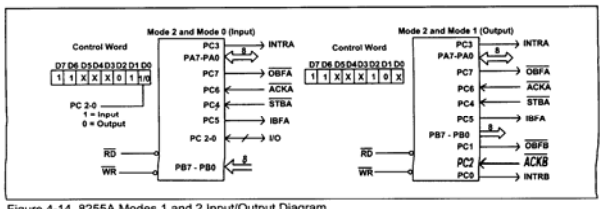



Figure 4-14. 8255A Modes 1 and 2 Input/Output Diagram
(Reprinted by permission of Intel Corporation, Copyright Intel Corp. 1983)



אם סיבית מספר 7 של אוגר הבקרה שווה לערך לוגי אפס אז כל סיבית של שער סי ניתן לתכנת כאפס או אחד על פי הטבלה.

If bit 7 of the control word is a logical 0 then each bit of the port C can be set or reset according to the table.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| | X | X | X | | | | |

Don't Care

Bit set/reset
1 = set
2 = reset

Bit set/reset
0 = ACTIVE

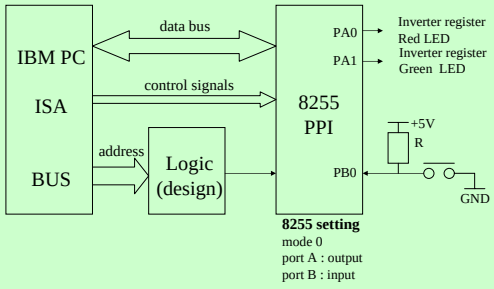
Bit SELECT

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | B0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | B1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | B2 |

**Examples:** If you want to set/reset bit 0 of port C then set D3 to D1 to 000.
Bit 1 of port C will be set/reset if you code 001 to D3 to D1.
Bit 6 of port C is set/reset if D3 to D1 is 110.



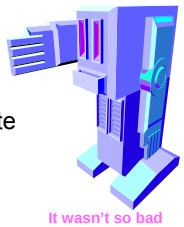
## Parallel Interface Block Diagram






## Intel 8255 PPI control

- I/O port address
  - 0x318 : 8255 PPI port A
  - 0x319 : 8255 PPI port B
  - 0x31A : 8255 PPI port C
  - 0x31B : 8255 PPI Control Byte
- Control Byte
  - 0x31B address data



It wasn't so bad after all



## Mode 2 Operation

- In mode 2, PORT A becomes a bidirectional bus supported by five handshaking signals
- The handshaking signals are the same as the five used in mode 1 for both input and output only then now apply to PORT A at the same time
- PORT B can be in either mode 0 or mode 1

# Mode 2 Input

- INPUT-peripheral sends data along $A_0$-$A_7$ to the 8255
  - from the standpoint of the peripheral this is an output mode
  - the peripheral sends an STB pulse to the 8255
  - when the data is latched, the IBF goes high
  - after STB returns to high with IBF still set, then INTR goes high
  - polling is possible at this point



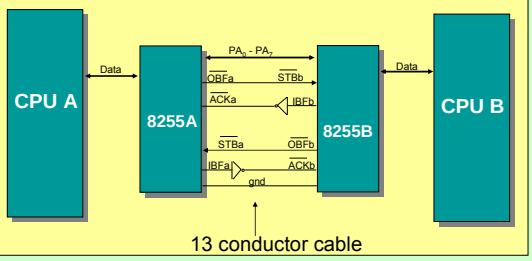
# Mode 2 Output

- OUTPUT - the 8255 sends data to the peripheral
  - when the data is loaded in the 8255 the OBF signal goes low
  - the peripheral acknowledges the low OBF by setting ACK low
  - on the falling edge of ACK, the 8255 places its data on PORT A
  - OBF returns high

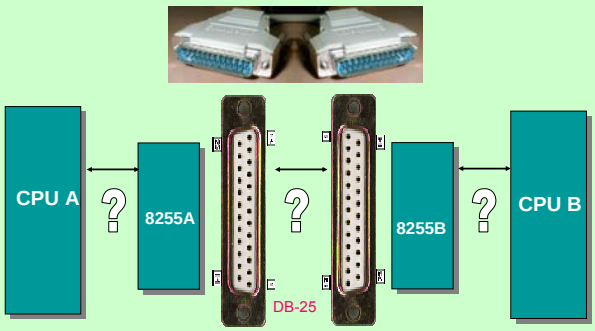


# Example 1

- Two 8255As may be used to interface two dissimilar computers over a common data bus








# Operation

- Communication from CPU a to CPU b
  - CPU a outputs its data to 8255A
  - the OBF signal goes low on the 8255A
  - As a result, the STB signal goes low into the 8255B
  - The 8255B responds with the IBF signal which appears to the 8255A as ACK
  - so, the 8255A places its data on PORT A
  - the 8255B reads the data, terminates the IBF forcing ACK high
  - the OBF signal of the 8255A goes high

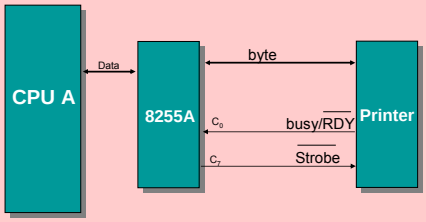


# Requirements

- For this computer-to-computer communication link to work the following is required:
  - an 8255 initialization routine
  - a polling routing to monitor IBF and OBF
  - a receiver program to read a data byte when IBF goes high
  - a transmitter program to output a data byte when OBF goes low

# Example 2

- PROBLEM: use the 8255 to interface to a parallel printer. Assume that a data table is stored at location DATA and the number of bytes to print is stored at location NUMB







# Code I

```
DATA = ADDR1      ;address of the data table
NUMB= ADDR2       ;address of the byte counter
;initialize the 8255
        MOV  AL, 81h          ;mode 0 control word
        OUT  0F3h,(AL)  ;send to the 8255 control port
        MOV  AL, 0FH          ;set bit 7 of C high
        OUT  0F3h,(AL)  ;send to the control port
;point at the data table and get the byte counter
        LXI  DX,DATA
        MOV  BL,(NUMB)
;check to see if the printer is ready
POLL: IN     AX,0F2h          ;read PORT C
        RAR  AX               ;move bit 0 into the carry
        JC   POLL             ;if printer is busy continue in loop
;printer is ready so send the data byte
        MOV  AL,(DX)          ;move the byte to A
        OUT  AL,0F0h          ;send it to PORT A
```




# Code II

```
;tell the printer that the data is ready
        MOV AL,0Eh            ;reset bit 7 of C
        OUT 0F3h,AL           ;send to PORT C
        INC AL          ;increment A (it now has 0Fh)
        OUT 0F3h,AL           ;set bit 7
;advance the data pointer and byte counter
        INC DX          ;point to the next byte
        DCR BL          ;decrement the byte counter
        JNZ POLL        ;if there are more bytes
        RET             ;else stop
```




# SDK-86 Parallel I/O Ports

**The parallel I/O ports consist of two 8255A programmable peripheral interfaces (PPI).**

**Each PPI contains three 8-bit I/O data ports and one write only control port. The three I/O ports are designated A, B, and C.**

**One 8255A, Port 1 or P1, is connected to the high data byte (d8-d15). The other 8255A, Port 2 or P2, is connected to the low data byte (d0 -d7).**

**All ports can be addressed individually with byte instructions (e.g., P1A or P2C) or corresponding ports can be addressed in pairs to form a 16-bit wide data port.**

**Figure 7-8, Sheet 5 of 9, of the textbook shows the SDK-86 parallel port. Note that the chip selects LOW PORT SELECT' and HIGH PORT SELECT' come from the I/O decode PROM.**

**Pins A1 and A0 on the 8255A are connected to A1 and A2 on the address bus.**

| A1 | A0 | |
|---|---|---|
| 0 | 0 | Port A |
| 0 | 1 | Port B |
| 1 | 0 | Port C |
| 1 | 1 | Control Port |




# 8255A

**Address Assignments**

| Port | Address | Port | Address |
|---|---|---|---|
| P2A | 0FFF8H | P1A | 0FFF9H |
| P2B | 0FFFAH | P1B | 0FFFBH |
| P2C | 0FFFCH | P1C | 0FFFDH |
| P2 Control | 0FFFEH | P1 Control | 0FFFFH |

**The 8255A has 24 programmable I/O pins - in two groups of 12. Figure 9-3 of the textbook shows the block diagram of the 8255A. Ports A and B are 8-bit ports, and Port C is split into two 4-bit ports.**

**The 8255A has three modes of operation.**

**Mode 0 -- Each 8-bit and 4-bit port can an input or output port.**

**Mode 1 -- Ports A or B can be an input or output port. Three lines of Port C are used for handshaking and interrupt control for each 8-bit port. If both ports are in Mode 1, only two pins in Port C are data pins.**

**Mode 2 -- Only Port A can be in mode 2. This is a bidirectional bus mode and requires five pins from Port C for handshaking. If A is in mode 2 and B is in mode 1, then there are no data lines left in Port C.**




# 8255A

**Figure 9-4 of the textbook shows a summary of the operating modes. Note that the pins in Port C Upper that are data pins when Port A is in mode 1 depends on whether Port A is input or output.**

**Figure 9-5 of the textbook shows the two control word formats. If D7 = 1, the control word is the mode-set control word. If D7 = 0, the control word is the Port C set/reset control word. This allows you to control single pins in Port C.**

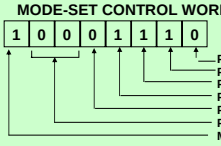
**Interrupt Control Functions**
**In mode 1 or 2, control signals can be used as interrupt request inputs. The interrupt request signals, generated from Port C, can be inhibited or enabled by setting or resetting a flip-flop using the single-bit set/reset function of Port C.**

**Figure 9-9 shows the Port C bits used for the interrupt request and for the interrupt enable flag (flip-flop).**

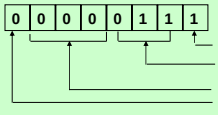
| | | Interrupt Req Pin | Interrupt Enable Bit |
|---|---|---|---|
| Mode 1 | Port A IN | PC3 | PC4 |
| | Port B IN | PC0 | PC2 |
| | Port A OUT | PC3 | PC6 |
| | Port B OUT | PC0 | PC2 |
| Mode 2 | Port A IN | PC3 | PC4 |
| | Port A OUT | PC3 | PC6 |

# 8255 CONTROL WORD EXAMPLES

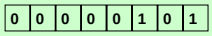
**MODE-SET CONTROL WORD**




**Three lines in Port C lower are used for control, leaving only 1 line for data.**




**Port C bit set/reset control word to set bit 3. Pin 3 is an output line that is set high.**




**Since Port B is in mode 1, this Port C bit set command sets PC2 which enables the interrupt request line on Port B.**



# SDK-86 8255A Programming

**The SDK-86 has two 8255A's, one on the lower half of the data bus and one on the upper half. 16-bit words are input or output by reading or writing to the even address. This means you write to port A, B, or C on each chip at the same time.**

**Example: Mode definition on the SDK-86**

```
MOV  DX,0FFFEH
MOV  AX,9090H          ; send the command word 90H to each 8255A
OUT  DX,AX             ; A - input, B - output, C - output
```