CHƯƠNG 15
Phép ghép 8031/51 với 8255

Như đã nói ở chương 14 trong quá trình nối ghép 8031/51 với bộ nhớ ngoài thì hai cổng P0 và P2 bị mất. Trong chương này chúng ta sẽ trình bày làm thế nào để mở rộng các cổng vào/ ra I/O của 8031/51 bằng việc nối nó tới chíp 8255.

15.1 Lập trình 8255.

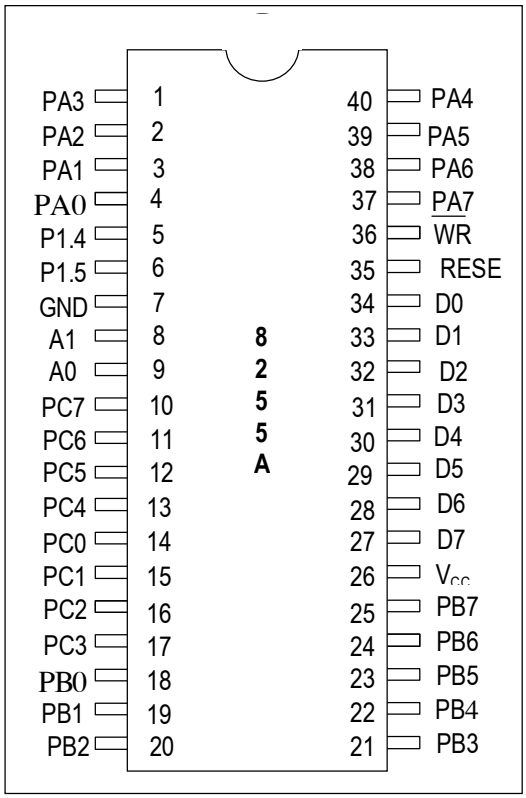
Trong mục này ta nghiên cứu 8255 như là một trong những chíp vào/ ra được sử dụng rộng rãi nhất. Trước hết ta mô tả những đặc tính của nó và sau đsó chỉ ra cách nối 8031/51 với 8255 như thế nào?

15.1 Lập trình 8255.

Trong mục này ta nghiên cứu 8255 như là một trong những chíp vào/ ra được sử dụng rộng rãi nhất. Trước hết ta mô tả những đặc tính của nó và sau đó chỉ ra cách nối 8031/51 với 8255 như thế nào?

15.1.1 Các đặc tính của 8255.

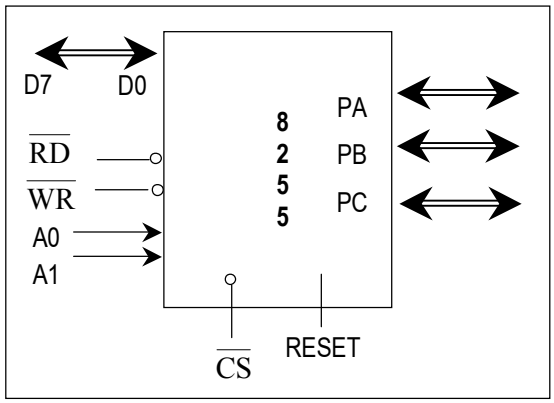
8255 là một chíp DIP 4 chân (xem hình 15.1). Nó có 3 cổng truy cập được riêng biệt. Các cổng đó có tên A, B và C đều là các cổng 8 bit. Các cổng này đều có thể lập trình như cổng đầu vào hoặc đầu ra riêng rẽ và có thể thay đổi một cách năng động. Ngoài ra, các cổng 8255 có khả năng bắt tay. Do vậy cho phép giao diện với các thiết bị khác cũng có giá trị tín hiệu bắt tay như các máy in chẳng hạn. Khả năng bắt tay của 8255 sẽ được bàn tới ở mục 15.3.



Hình 15.1: Chíp 8255.

15.1.1.1 Các chân PA0 - PA7 (cổng A).

Cả 8 bít của cổng A PA0 - PA7 có thể được lập trình như 8 bit đầu vào hoặc 8 bit đầu ra hoặc cả 8 bít hai chiều vào/ ra.S



Hình 15.2: Sơ đồ khối của 8255.

15.1.1.2 Các chân PB0 - PB7 (cổng B).

Cả 8 bit của cổng B có thể được lập trình hoặc như 8 bit đầu vào hoặc 8 bit đầu ra hoặc cả 8 bit hai chiều vào/ ra.

15.1.1.3 Các chân PC0 - PC7 (cổng C).

Tất cả 8 bit của cổng C (PC0 - PC7) đều có thể được lập trình như các bit đầu vào hoặc các bit đầu ra. 8 bit này cũng có thể được chia làm hai phần: Các bit cao (PC4 - PC7) là CU và các bit thấp (PC0 - PC3) là CL. Mỗi phần có thể được dùng hoặc làm đầu vào hoặc làm đầu ra. Ngoài ra từng bit của cổng C từ PC0 - PC7 cũng có thể được lập trình riêng rẽ.

15.1.1.4 Các chân $\overline{RD}$ và $\overline{WR}$.

Đây là hai tín hiệu điều khiển tích cực mức thấp tới 8255 được nối tới các chân dữ liệu $\overline{RD}$ và $\overline{WR}$ từ 8031/51 được nối tới các chân đầu vào này.

15.1.1.5 Các chân dữ liệu D0 - D7.

Các chân dữ liệu D0 - D7 của 8255 được nối tới các chân dữ liệu của bộ vi điều khiển để cho phép nó gửi dữ liệu qua lại giữa bộ vi điều khiển và chíp 8255.

15.1.1.6 Chân RESET.

Đây là đầu vào tín hiệu tích cực mức cao tới 8255 được dùng để xoá thanh ghi điều khiển. Khi chân RESET được kích hoạt thì tất cả các cổng được khởi tạo lại như các cổng vào. Trong nhiều thiết kế thì chân này được nối tới đầu ra RESET của bus hệ thống hoặc được nối tới đất để không kích hoạt nó. Cũng như tất cả các chân đầu vào của IC thì nó cũng có thể để hở.

15.1.1.7 Các chân A0, A1 và $\overline{CS}$.

Trong khi $\overline{CS}$ chọn toàn bộ chíp thì A0 và A1 lại chọn các cổng riêng biệt. Các chân này được dùng để truy cập các cổng A, B, C hoặc thanh ghi điều khiển theo bảng 15.1. Lưu ý $\overline{CS}$ là tích cực mức thấp.

15.1.2 Chọn chế độ của 8255.

Trong khi các cổng A, B và C được dùng để nhập và xuất dữ liệu thì thanh ghi điều khiển phải được lập trình để chọn chế độ làm việc của các cổng này. Các cổng của 8255 có thể được lập trình theo một chế độ bất kỳ dưới đây.

1. Chế độ 0 (Mode0): Đây là chế độ vào/ ra đơn giản. Ở chế độ này các cổng A, B CL và CU có thể được lập trình như đầu vào hoặc đầu ra. Trong chế độ này thì tất cả các bit hoặc là đầu vào hoặc là đầu ra. Hay nói cách khác là không có điều khiển theo từng bit riêng rẽ như ta đã thấy ở các cổng P0 - P3 của 8051. Vì đa phần các ứng dụng liên quan đến 8255 đều sử dụng chế độ vào/ ra đơn giản này nên ta sẽ tập chung đi sâu vào chế độ này.

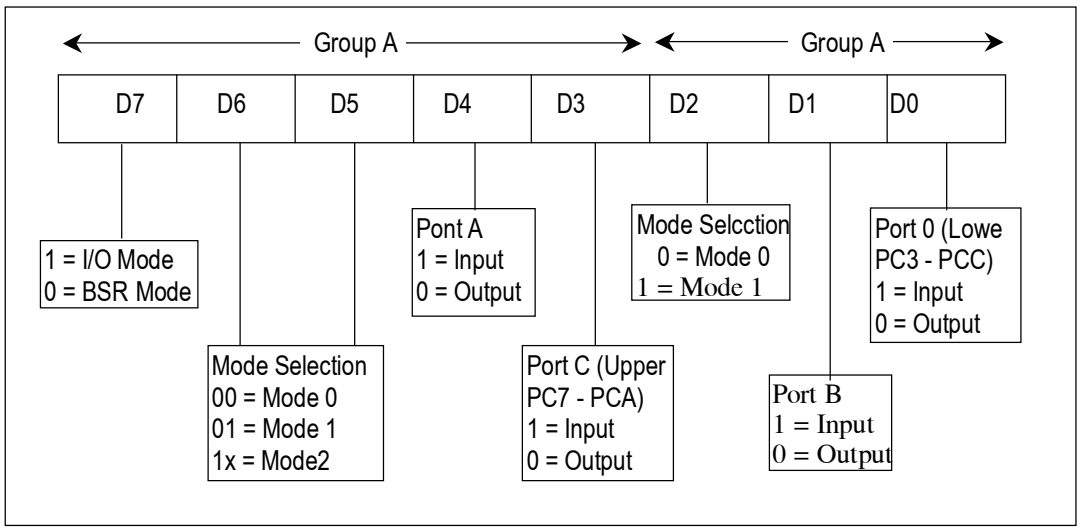
2. Chế độ 1 (Mode1): Trong chế độ này các cổng A và B có thể được dùng như các cổng đầu vào hoặc đầu ra với các khả năng bắt tay. Tín hiệu bắt tay được cấp bởi các bit của cổng C (sẽ được trình bày ở mục 15.3).

3. Chế độ 2 (Mode2): Trong chế độ này cổng A có thể được dùng như cổng vào/ ra hai chiều với khả năng bắt tay và các tín hiệu bắt tay được cấp bởi các bit cổng C. Cổng B có thể được dùng như ở chế độ vào/ ra đơn giản hoặc ở chế độ có bắt tay Mode1. Chế độ này sẽ không được trình bày trong tài liệu này.

Chế độ BSR: Đây là chế độ thiết lập/ xoá bit (Bit Set/ Reset). ở chế độ này chỉ có những bit riêng rẽ của cổng C có thể được lập trình (sẽ được trình bày ở mục 15.3).

Bảng 15.1: Chọn cổng của 8255.

$\overline{CS}$	A1	A0	Chọn cổng
0	0	0	Cổng A
0	0	1	Cổng B
0	1	0	Cổng C
0	1	1	Thanh ghi điều khiển
1	x	X	8255 không được chọn



Hình 15.3: Định dạng từ điều khiển của 8255 (chế độ vào/ ra).

15.1.3 Lập trình chế độ vào/ ra đơn giản.

Hãng Intel gọi chế độ 0 là chế độ vào/ ra cơ sở. Một thuật ngữ được dùng chung hơn là vào/ ra đơn giản. Trong chế độ này thì một cổng bất kỳ trong A, B, C được lập trình như là cổng đầu vào hoặc cổng đầu ra. Cần lưu ý rằng trong chế độ này một cổng đã cho không thể vừa làm đầu vào lại vừa làm đầu ra cùng một lúc.

Ví dụ 15.1:

Hãy tìm từ điều khiển của 8255 cho các cấu hình sau:

Tất cả các cổng A, B và C đều là các cổng đầu ra (chế độ 0).

PA là đầu vào, PB là đầu ra, PCL bằng đầu vào và PCH bằng đầu ra.

Lời giải:

Từ hình 15.3 ta tìm được:

a) 1000 0000 = 80H; b) 1001 000 = 90H

15.1.4 Nối ghép 8031/51 với 8255.

Chíp 8255 được lập trình một trong bốn chế độ vừa trình bày ở trên bằng cách gửi một byte (hãng Intel gọi là một từ điều khiển) tới thanh ghi điều khiển của 8255. Trước hết chúng ta phải tìm ra các địa chỉ cổng được gán cho mỗi cổng A, B, C và thanh ghi điều khiển. Đây được gọi là ánh xạ cổng vào/ ra (mapping).

Như có thể nhìn thấy từ hình 15.4 thì 8255 được nối tới một 8031/51 như thế nó là bộ nhớ RAM. Để việc sử dụng các tín hiệu $\overline{RD}$ và $\overline{WR}$. Phương pháp nối một chíp vào/ ra bộ nhớ vì nó được ánh xạ vào không gian bộ nhớ. Hay nói cách khác, ta sử dụng không gian bộ nhớ để truy cập các thiết bị vào/ ra. Vì ly??? do này mà ta dùng lệnh MOVX để truy cập RAM và ROM. Đối với một 8255 được nối tới 8031/51 thì ta cũng phải dùng lệnh MOVX để truyền thông với nó. Điều này được thể hiện trên ví dụ 15.2.

Ví dụ 15.2:

Đối với hình 15.4:

a) Hãy tìm các địa chỉ vào/ ra được gán cho cổng A, B, C và thanh ghi điều khiển.
b) Hãy lập trình 8255 cho các cổng A, B và C thành các cổng đầu ra.
c) Viết một chương trình để gửi 55H và AAH đến cổng liên tục.

Lời giải:

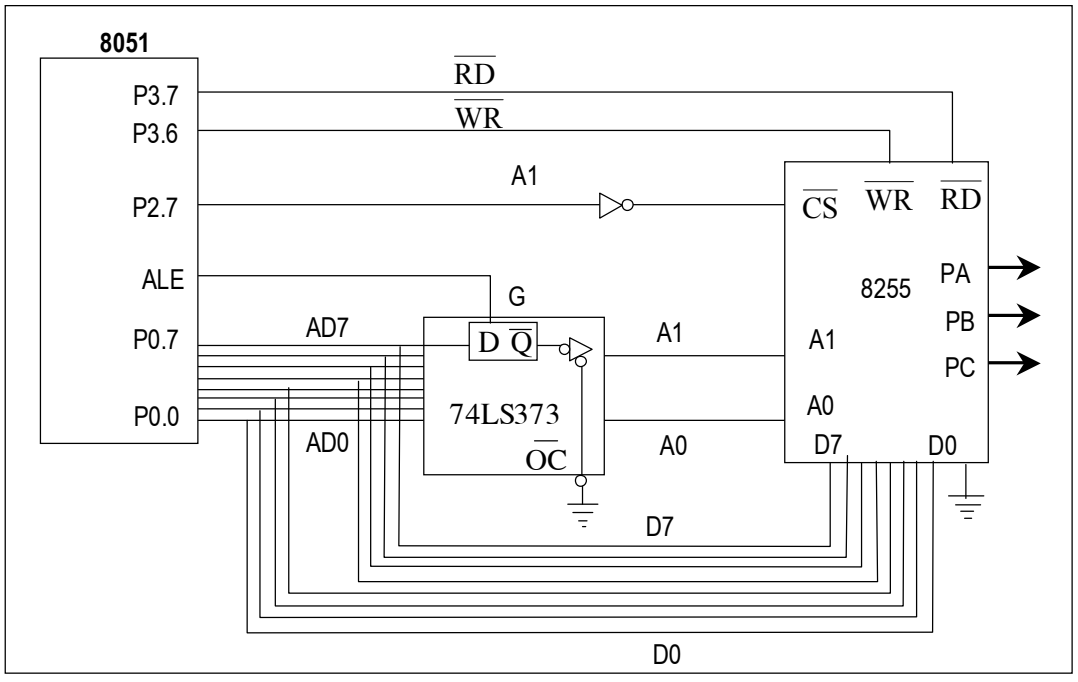
a) Địa chỉ cơ sở dành cho 8255 như sau:

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
x	1	x	x	x	x	x	x	x	x	x	x	x	X	0	0	=4000HPA
x	1	x	x	x	x	x	x	x	x	x	x	x	X	0	1	=4000HPB
x	1	x	x	x	x	x	x	x	x	x	x	x	X	1	0	=4000HPC
x	1	x	x	x	x	x	x	x	x	x	x	x	X	1	1	=4000HCR

b) Byte (từ) điều khiển cho tất cả các cổng như đầu ra là 80H như được tính ở ví dụ 15.1.

c)

```
        MOV     A, #80H         ; Từ điển khiển
        MOV     DPTR, # 4003H   ; Nạp địa chỉ cổng của thanh ghi điều khiển
        MOVX    @DPTR, A        ; Xuất từ điển khiển
        MOV     A, # 55H        ; Gán A = 55
AGAIN:  MOV     DPTR, # 4000H   ; Địa chỉ cổng PA
        MOVX    @DPTR, A        ; Lấy các bit cổng PA
        INC     DPTR            ; Địa chỉ cổng PB
        MOVX    @DPTR, A        ; Lấy các bít cổng PB
        INC     DPTR            ; Địa chỉ cổng PC
        MOVX    @DPTR, A        ; Lấy các bít cổng PC
        CPL     A               ; Lấy các bít thanh ghi A
        ACALL   DELAY           ; Chờ
        SJMP    AGAIN           ; Tiếp tục
```



Hình 15.4: Nối ghép 8051 với 8255 cho ví dụ 15.2.

Ví dụ 15.3:

Đối với hình 15.5:

a) Tìm các địa chỉ cổng vào ra được gán cho các cổng A, B, C và thanhg ghi điều khiển.

b) Tìm byte điều khiển đối với PA bằng đầu vào, PB bằng đầu ra, PC bằng đầu ra

c) Viết một chương trình để nhận dữ liệu từ PA gửi nó đến cả cổng B và cổng C.

Lời giải:

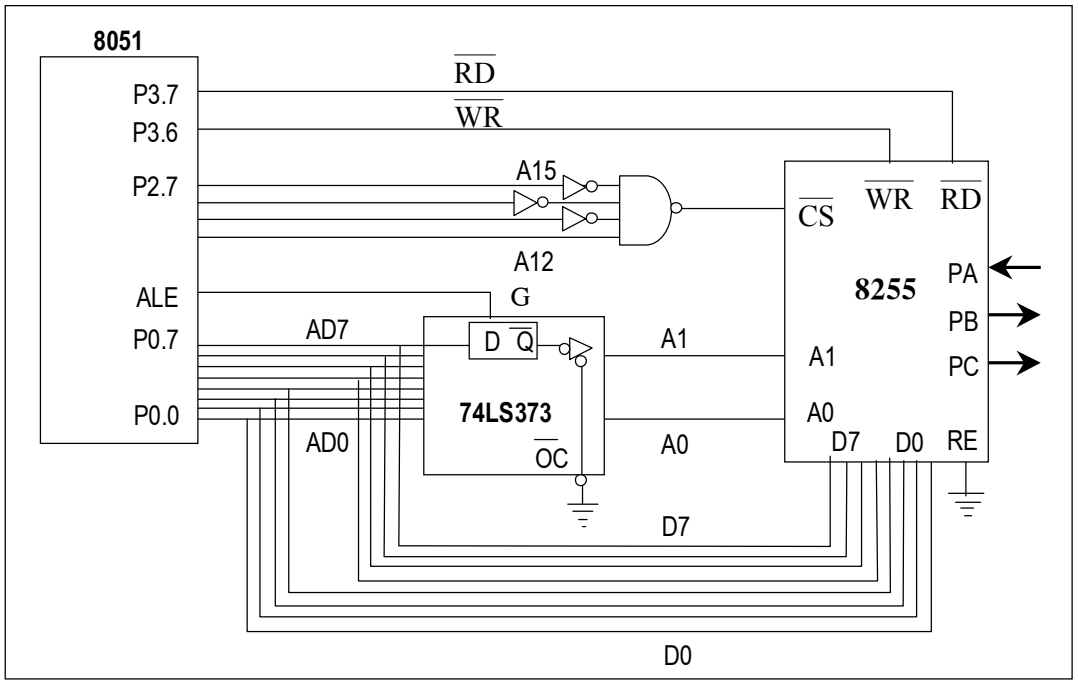
a) Giả sử tất các các bit không dùng đến là 0 thì địa chỉ cổng cơ sở cho 8255 là 1000H. Do vậy ta có:

1000H là PA; 1001H là PB; 1002H là PC và 1003H là thanh ghi điều khiển.

b) Từ điều khiển cho trường hợp này là 10010000 hay 90H.

c)

```
MOV     A, #90H ; PA là đầu vào, PB là đầu ra, PC là đầu ra
MOV     DPTR, #1003H    ; Nạp địa chỉ cổng của thanh ghi điều khiển
MOVX    @DPTR, A        ; Xuất từ điều khiển
MOV     DPTR, #1000H    ; Địa chỉ PA
MOVX    A, @DPTR        ; Nhận dữ liệu từ PA
INC     DPTR            ; Địa chỉ PB
MOVX    @DPTR, A        ; Gửi dữ liệu ra PB
INC     DPTR            ; Địa chỉ PC
MOVX    @DPTR, A        ; Gửi dữ liệu ra PC
```



Hình 15.5: Nối ghép 8051 tới 8255 cho ví dụ 15.3.

Đối với ví dụ 15.3 ta nên dùng chỉ lệnh EQU cho địa chỉ các cổng A, B, C và thanh ghi điều khiển CNTPORT như sau:

```
APORT  EQU    1000H
BPORT  EQU    1001H
CPORT  EQU    1002H
CNTPORT        EQU     1003H

MOV             A, #90H                 ; PA là đầu vào, PB là đầu ra, PC là đầu ra
MOV             DPTR, #CNTPORT          ; Nạp địa chỉ của cổng thanh ghi điều khiển
```

```
        MOVX        @DPTR, A            ; Xuất từ điều khiển
        MOV         DPTR, #CNTPORT      ; Địa chỉ PA
        MOVX        DPTR, APORT         ; Nhận dữ liệu PA
        INC         A, @DPTR            ; Địa chỉ PB
        MOVX        DPTR                ; Gửi dữ liệu ra PB
        INC         DPTR                ; Địa chỉ PC
        MOVX        DPTR, A       ; Gửi dữ liệu ra PC
```

hoặc có thể viết lại như sau:

```
CONTRBYT    EQU     90H                 Xác định PA đầu vào, PB và PC đầu ra
BAS8255P    EQU     1000H               ; Địa chỉ cơ sở của 8255

            MOV     A, #CONTRBYT
            MOV     DPTR, #BAS8255P+3   ; Nạp địa chỉ cổng C
            MOVX    @DPTR, A            ; Xuất từ điều khiển
            MOV     DPTR, #BASS8255P    ; Địa chỉ cổng A
            ...
```

Để ý trong ví dụ 15.2 và 15.3 ta đã sử dụng thanh ghi DPTR vì địa chỉ cơ sở gán cho 8255 là 16 bit. Nếu địa chỉ cơ sở dành cho 8255 là 8 bit, ta có thể sử dụng các lệnh "MOVX A, @R0" và "MOVX @R0, A" trong đó R0 (hoặc R1) giữ địa chỉ cổng 8 bit của cổng. Xem ví dụ 15.4, chú ý rằng trong ví dụ 15.4 ta sử dụng một cổng lôgíc đơn giản để giải mã địa chỉa cho 8255. Đối với hệ thống có nhiều 8255 ta có thể sử dụng 74LS138 để giải mã như sẽ trình bày ở ví dụ 15.5.

15.1.5 Các bí danh của địa chỉ (Addren Alias).

Trong các ví dụ 15.4 và 15.4 ta giải mã các bít địa chỉ A0 - A7, tuy nhiên trong ví dụ 15.3 và 15.2 ta đã giải mã một phần các địa chỉ cao của A8 - A15. Việc giải mã từng phần này dẫn đến cái gọi là các bí danh của địa chỉ (Address Aliases). Hay nói cách khác, cùng cổng vật lý giống nhau có các địa chỉ khác nhau, do vậy cùng một cổng mà được biết với các tên khác nhau. Trong ví dụ 15.2 và 15.3 ta có thể thay đổi tốt x thành các tổ hợp các số 1 và 0 khác nhau thành các địa chỉ khác nhau, song về thực chất chúng tham chiếu đến cùng một cổng vật lý. Trong tài liệu thuyết minh phần cứng của mình chúng ta cần phải bảo đảm ghi chú đầy đủ các bí danh địa chỉa nếu có sao cho mọi người dùng biết được các địa chỉ có sẵn để họ có thể mở rộng hệ thống.

Ví dụ 15.4:

Cho hình 15.6:

a) Hãy tìm các địa chỉ cổng vào/ ra được gán cho các cổng A, B, C và thanh ghi điều khiển.

b) Tìm từ điều khiển cho trường hợp PA là đầu ra, PB là đầu vào, PC - PC3 là đầu vào và CP4 - CP7 là đầu ra.

c) Viết một chương trình để nhận dữ liệu từ PB và gửi nó ra PA. Ngoài ra, dữ liệu từ PC1 được gửi đến CPU.

Lời giải:

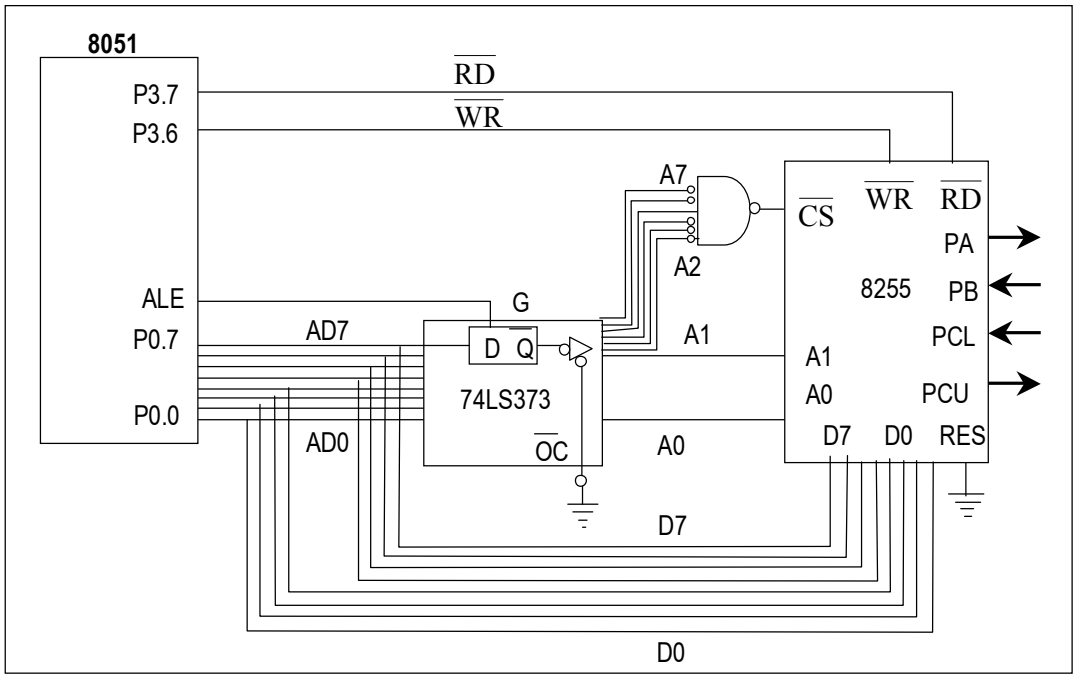
a) Các địa chỉ cổng được tìm thấy như sau:

BB	$\overline{CS}$	A1	A0	Địa chỉ	Cổng
0010	00	0	0	20H	Cổng A
0010	00	0	1	21H	Cổng B
0010	00	1	0	22H	Cổng C
0010	00	1	1	23H	Thanh ghi điều khiển

b) Từ điều khiển là 10000011 hay 83H.

c)

```
CONTRBYT    EQU     83H         ; PA là đầu ra, PB,PCL là đầu vào
APORT       EQU     20H
BPORT       EQU     21H
CPORT       EQU     22H
CNTPORT     EQU     23H
            ...
            MOV     A, #CONTRBYT
            MOV     A, #CONTRBYT  ; PA, PCU là đầu ra, PB và PCL là đầu vào
            MOV     R0, #CNTPORT  ; Nạp địa chỉ của cổng thanh ghi điều khiển
            MOVX    @R0, A        ; Xuất từ điều khiển
            MOV     R0, #BPORT    ; Nạp địa chỉ PB
            MOVX    A, @R0        ; Đọc PB
            DEC     R0            ; Chỉ đến PA (20H)
            MOVX    @R0, A        ; Gửi nó đến PA
            MOV     R0, #CPORT    ; Nạp địa chỉ PC
            MOVX    A, @R0        ; Đọc PCL
            ANL     A, #0FH       ; Che phần cao
            SWAP    A             ; Trao đổi phần cao và thấp
            MOVX    @R0, A        ; Gửi đến PCU
```



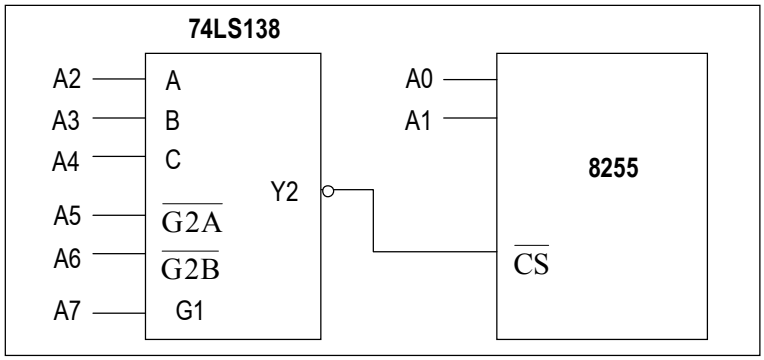
Hình 15.6: Nối ghép 8051 với 8255 cho ví dụ 15.4.

Ví dụ 15.5:

Hãy tìm địa chỉ cơ sở cho 8255 trên hình 15.7.

Lời giải:

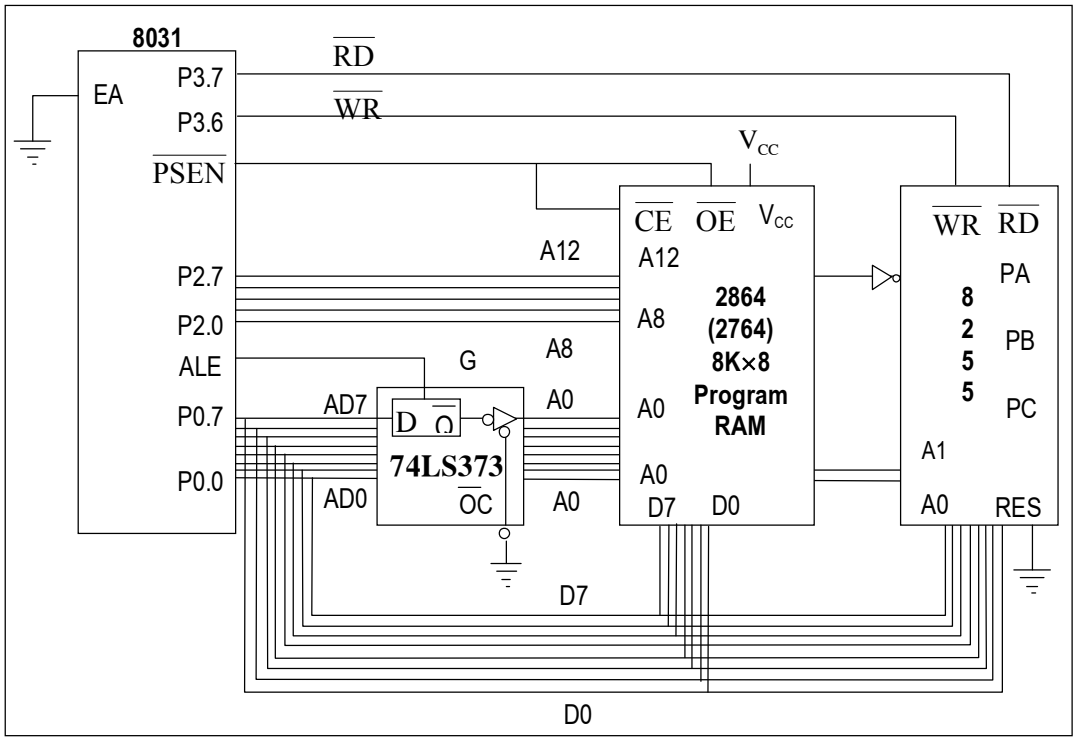
GA	$\overline{G2B}$	$\overline{G2A}$	C	B	A			Địa chỉ
A7	A6	A5	A4	A3	A2	A1	A0	
1	0	0	0	1	0	0	0	88H



Hình 15.7: Giải mã địa chỉ của 8255 sử dụng 74LS138.

15.1.6 Hệ 8031 với 8255.

Trong một hệ thống dựa trên 8031 mà bộ nhớ chương trình ROM ngoài là một sự bắt buộc tuyệt đối thì sử dụng một 8255 là rất được trào đón. Điều này là do một thực tế là trong giải trình phối ghép 8031 với bộ nhớ chương trình ROM ngoài ta bị mất hai cổng P0 và P2 và chỉ còn lại duy nhất cổng P1. Do vậy, việc nối với một 8255 là cách tốt nhất để có thêm một số cổng. Điều này được chỉ ra trên hình 15.8.

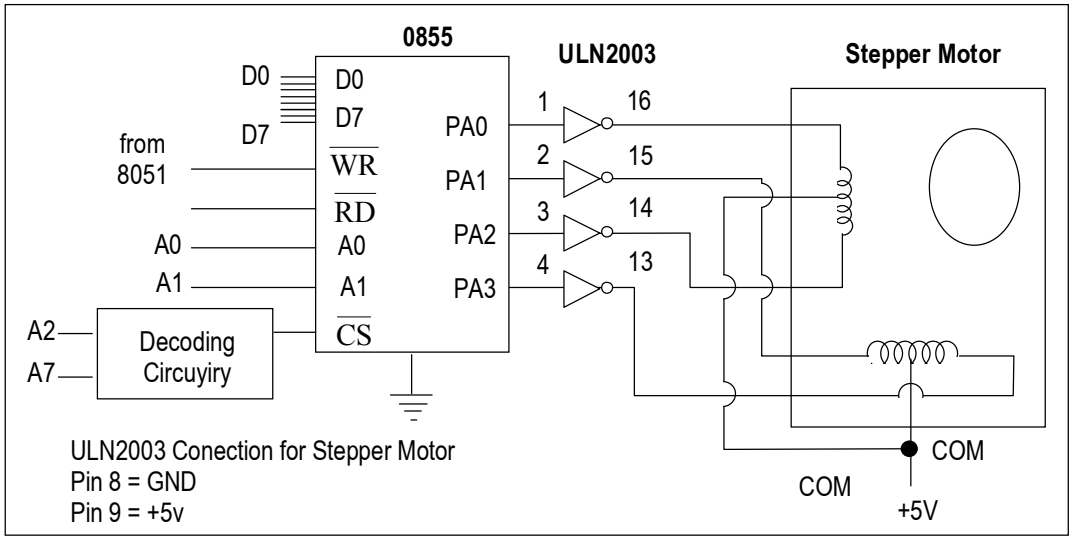


Hình 15.8: Nối 8031 tới một ROM chương trình ngoài và 8255.

15.2 Nối ghép với thế giới thực.

15.2.1 Phối ghép 8255 với động cơ bước.

Chương 13 đã nói chi tiết về phối ghép động cơ bước với 8051, ở đây ta trình bày nối ghép động cơ bước tới 8255 và lập trình (xem hình 15.9).

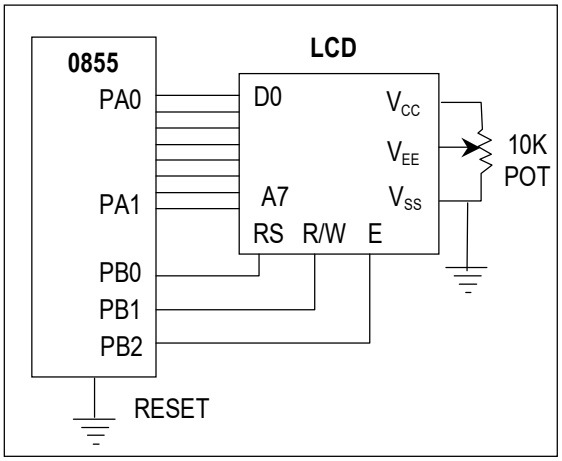


Hình 15.9: Nối ghép 8255 với một động cơ bước.
Chương trình cho sơ đồ nối ghép này như sau:

```
        MOV     A, #80H         ; Chọn từ điều khiển để PA là đầu ra
        MOV     R1, #CRPORT     ; Địa chỉ cổng thanh ghi điều khiển
        MOVX    @R1, A          ; Cấu hình cho PA đầu ra
        MOV     R1, #APORT      ; Nạp địa chỉ cổng PA
        MOV     A, #66H         ; Gán A = 66H, chuyển xung của động cơ bước
AGAIN:  MOVX    @R1, A          ; Xuất chuỗi động cơ đến PA
        RR      A               ; Quay chuỗi theo chiều kim đồng hồ
        ACALL   DELAY           ; Chờ
        SJMP    AGAIN
```

15.2.2 Phối ghép 8255 với LCD.

Chương trình 15.1 trình bày cách xuất các lệnh và dữ liệu tới một LCD được nối tới 8255 theo sơ đồ hình 15.10. Trong chương trình 15.1 ta phải đặt một độ trễ trước mỗi lần xuất thông tin bất kỳ (lệnh hoặc dữ liệu) tới LCD. Một cách tốt hơn là kiểm tra cờ bận trước khi xuất bất kỳ thứ gì tới LCD như đã nói ở chương 12. Chương trình 15.2 lặp lại chương trình 15.1 có sử dụng kiểm tra cờ bận. Để ý rằng lúc này không cần thời gian giữ chậm như ở vị trí 15.1.



Hình 5.10: Nối ghép 8255 với LCD.

Chương 15.1:

```
; Ghi các lệnh và dữ liệu tới LCD không có kiểm tra cờ bận.
; Giả sử PA của 8255 được nối tới D0 - D7 của LCD và
; IB - RS, PB1 = R/W, PB2 = E để nối các chân điều khiển LCD
        MOV     A, #80H         ; Đặt tất cả các cổng 8255 là đầu ra
        MOV     R0, #CNTPORT    ; Nẹp địa chỉ thanh ghi điều khiển
        MOVX    @R0, A          ; Xuất từ điều khiển
        MOV     A, #38H         ; Cấu hình LCD có hai dòng và ma trận 5×7
```

```
                ACALL       CMDWRT      ; Ghi lệnh ra LCD
                ACALL       DELAY       ; Chờ đến lần xuất kế tiếp (2ms)
                MOV         A, # 0EH    ; Bật con trỏ cho LCD
                ACALL       CMDWRT      ; Ghi lệnh này ra LCD
                ACALL       DELAY       ; Chờ lần xuất kế tiếp
                MOV         A, # 01H    ; Xoá LCD
                ACALL       CMDWRT      ; Ghi lệnh này ra LCD
                ACALL       DELAY       ; Dịch con trỏ sang phải
                MOV         A, # 06     ; Ghi lệnh này ra LCD
                ACALL       CMDWRT      ; Chờ lần xuất sau
                ACALL       DELAY       ; Ghi lệnh này ra LCD
                ...                     ; v.v... cho tất cả mọi lệnh LCD
                MOV         A, # 'N'    ; Hiển thị dữ liệu ra (chữ N)
                ACALL       DATAWRT     ; Gửi dữ liệu ra LCD để hiển thị
                ACALL       DELAY       ; Chờ lần xuất sau
                MOV         A, # '0'    ; Hiển thị chữ "0"
                ACALL       DATAWRT      ; Gửi ra LCD để hiển thị
                ACALL       DELAY       ; Chờ lần xuất sau
                ...                     ; v.v... cho các dữ liệu khác
; Chương trình con ghi lệnh CMDWRT ra LCD
CMDWRT:         MOV         R0, # APORT     ; Nạp địa chỉ cổng A
                MOVX        @R0, A          ; Xuất thông tin tới chân dữ liệu của LCD
                MOV         R0, # BPORT     ; Nạp địa chỉ cổng B
                MOV         A, # 00000100B  ; RS=0, R/W=1, E=1 cho xung cao xuống thấp

                MOVX        @R0, A          ; Kích hoạt các chânRS, R/W, E của LCD
                NOP                         ; Tạo độ xung cho chân E
                NOP
                MOV         A, # 00000000B  ; RS=0, R/W=1, E=1 cho xung cao xuống thấp

                MOVX        @R0, A          ; Chốt thông tin trên chân dữ liệu của LCD
                RET
; Chương trình con ghi lệnh DATAWRT ghi dữ liệu ra LCD.
CMDWRT:         MOV         R0, # APORT     ; Nạp địa chỉ cổng A
                MOVX        @R0, A          ; Xuất thông tin tới chân dữ liệu của LCD
                MOV         R0, # BPORT     ; Đặt RS=1, R/W=0, E=0 cho xung cao xuống thấp
                MOV         A, # 00000101B  ; Kích hoạt các chân RS, R/W, E
                MOVX        @R0, A          ; Tạo độ xung cho chân E
                NOP
                NOP
                MOV         A, # 00000001B  ; Đặt RS=1, R/W=0, E=0 cho xung cao xuống thấp
                MOVX        @RC, A          ; Chốt thông tin trên chân dữ liệu của LCD
                RET
```

Chương trình 15.2:

```
; Ghi các lệnh và dữ liệu tới LCD có sử dụng kiểm tra cờ bận.
; Giả sử PA của 8255 được nối tới D0 - D7 của LCD và
; PB0 = RS, PB1 = R/W, PB2 = E đối với 8255 tới các chân điều khiển LCD
                MOV         A, #80H         ; Đặt tất cả các cổng 8255 là đầu ra
                MOV         R0, #CNTPORT    ; Nạp địa chỉ thanh ghi điều khiển
                MOVX        @R0, A          ; Xuất từ điều khiển
                MOV         A, #38H         ; Chọn LCD có hai dòng và ma trận 5×7
                ACALL       NMDWRT          ; Ghi lệnh ra LCD
                MOV         A, # 0EH        ; Lệnh của LCD cho con trỏ bật
                ACALL       NMDWRT          ; Ghi lệnh ra LCD
                MOV         A, # 01H        ; Xoá LCD
```

```
                ACALL       NMDWRT      ; Ghi lệnh ra LCD
                MOV         A, # 06     ; Lệnh dịch con trỏ sang phải
                ACALL       CMDWRT      ; Ghi lệnh ra LCD
                ...                     ; v.v... cho tất cả mọi lệnh LCD
                MOV         A, # 'N'    ; Hiển thị dữ liệu ra (chữ N)
                ACALL       NCMDWRT     ; Gửi dữ liệu ra LCD để hiển thị
                MOV         A, # '0'    ; Hiển thị chữ "0"
                ACALL       NDADWRT     ; Gửi ra LCD để hiển thị
                ...                     ; v.v... cho các dữ liệu khác
; Chương trình con ghi lệnh NCMDWRT có hiển thị cờ bận
NCMDWRT:        MOV         R2,  A      ; Lưu giá trị thanh ghi A
                MOV         A, #90H     ; Đặt PA là cổng đầu vào để đọc trạng thái LCD
                MOV         R0, # CNTPORT ; Nạp địa chỉ thanh ghi điều khiển
                MOVX        @R0, A      ; Đặt PA đầu vào, PB đầu ra
                MOV         A, # 00000110B ; RS=0, R/W=1, E=1 đọc lệnh
                MOV         @R0, BPORT  ; Nạp địa chỉ cổng B
                MOVX        R0, A       ; RS=0, R/W=1 cho các chân RD và RS

                MOV         R0, APORT   ; Nạp địa chỉ cổng A
READY:          MOVX        @R0         ; Đọc thanh ghi lệnh
                RLC         A           ; Chuyển D7 (cờ bận) vào bit nhớ carry
                JC          READY       ; Chờ cho đến khi LCD sẵn sàng
                MOV         A, #80H     ; Đặt lại PA, PB thành đầu ra
                MOV         R0, #CNTPORT ; Nạp địa chỉ cổng điều khiển
                MOVX        @R0, A      ; Xuất từ điều khiển tới 8255
                MOV         A, R2       ; Nhận giá trị trả lại  tới LCD
                MOV         R0, #APORT  ; Nạp địa chỉ cổng A
                MOVX        @R0, A      ; Xuất thông tin tới các chân dữ liệu của LCD
                MOV         R0, #BPORT  ; Nạp địa chỉ cổng B
                MOV         A, #00000100B ; Đặt RS=0, R/W=0, E=1 cho xung thấp lên cao
                MOVX        @R0, A      ; Kích hoạt RS, R/W, E của LCD
                NOP                     ; Tạo độ rộng xung của chân E
                NOP
                MOV         A, #00000000B ; Đặt RS=0, R/W=0, E=0 cho xung cao xuống thấp
                MOVX        @R0, A      ; Chốt thông tin ở chân dữ liệu LCD
                RET
; Chương trình con ghi dữ liệu mới NDATAWRT sử dụng cờ bận
NCMDWRT:        MOV         R2,  A      ; Lưu giá trị thanh ghi A
                MOV         A, #90H     ; Đặt PA là cổng đầu vào để đọc trạng thái LCD
                MOV         R0, # CNTPORT ; Nạp địa chỉ thanh ghi điều khiển
                MOVX        @R0, A      ; Đặt PA đầu vào, PB đầu ra
                MOV         A, # 00000110B ; RS=0, R/W=1, E=1 đọc lệnh
                MOV         @R0, BPORT  ; Nạp địa chỉ cổng B
                MOVX        R0, A       ; RS=0, R/W=1 cho các chân RD và RS

                MOV         R0, APORT   ; Nạp địa chỉ cổng A
READY:          MOVX        @R0         ; Đọc thanh ghi lệnh
                RLC         A           ; Chuyển D7 (cờ bận) vào bit nhớ carry
                JC          READY       ; Chờ cho đến khi LCD sẵn sàng
                MOV         A, #80H     ; Đặt lại PA, PB thành đầu ra
                MOV         R0, #CNTPORT ; Nạp địa chỉ cổng điều khiển
                MOVX        @R0, A      ; Xuất từ điều khiển tới 8255
                MOV         A, R2       ; Nhận giá trị trả lại  tới LCD
                MOV         R0, #APORT  ; Nạp địa chỉ cổng A
                MOVX        @R0, A      ; Xuất thông tin tới các chân dữ liệu của LCD
                MOV         R0, #BPORT  ; Nạp địa chỉ cổng B
                MOV         A, #00000101B ; Đặt RS=1, R/W=0, E=1 cho xung thấp lên cao
```

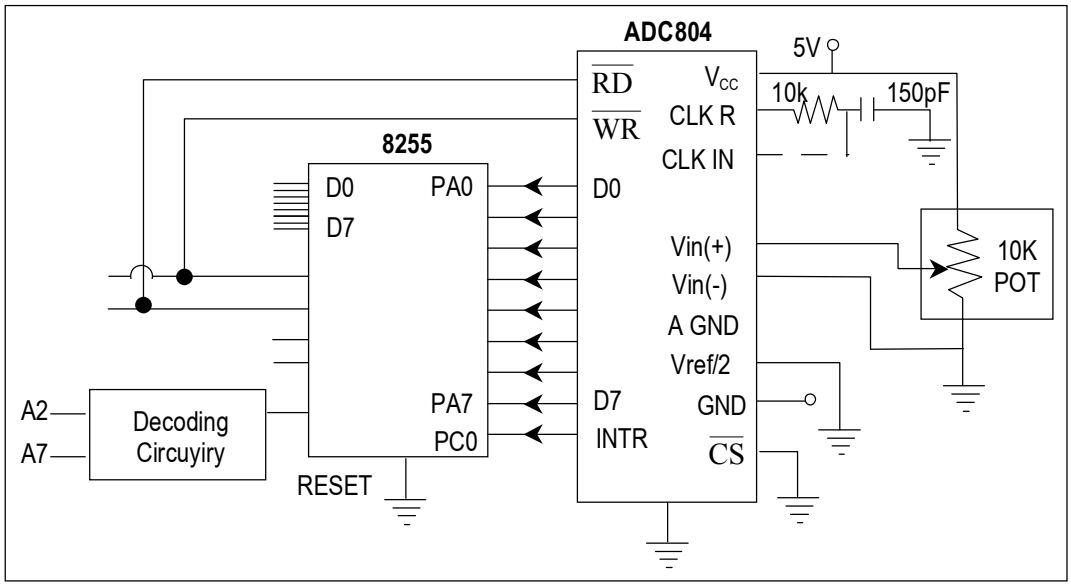
```
        MOVX    @R0, A          ; Kích hoạt RS, R/W, E của LCD
        NOP                     ; Tạo độ rộng xung của chân E
        NOP
        MOV     A, #00000001B   ; Đặt RS=1, R/W=0, E=0 cho xung cao xuống thấp
        MOVX    @R0, A          ; Chốt thông tin ở chân dữ liệu LCD
        RET
```

15.2.3 Nối ghép ADC tới 8255.

Các bộ ADC đã được trình bày ở chương 12. Dưới đây một chương trình chỉ một bộ ADC được nối tới 8255 theo sơ đồ cho trên hình 115.11.

```
        MOV     A, #80H         ; Từ điều khiển với PA = đầu ra và PC  = đầu vào
        MOV     R1, #CRPORT     ; Nạp địa chỉ cổng điều khiển
        MOVX    @R1, A          ; Đặt PA = đầu ra và PC = đầu vào
BACK:   MOV     R1, #CPORT      ; Nạp địa chỉ cổng C
        MOVX    A, @R1          ; Đọc địa chỉ cổng C để xem ADC đã sẵn sàng chưa
        ANL     A,, #00000001B  ; Che tất cả các bít cổng C để xem ADC đã sẵn
sàng chưa
        JNZ     BACK            ; Giữ hiển thị PC0 che EOC
                                ; Kết thúc hội thoại và bây giờ nhận dữ liệu của ADC
        MOV     R1, #APORT      ; Nạp địa chỉ PA
        MOVX    A, @R1          ; A = đầu vào dữ liệu tương tự
```

Cho đến đây ta đã được trao đổi chế độ vào/ ra đơn giản của 8255 và trình bày nhiều ví dụ về nó. Ta xét tiếp các chế độ khác.

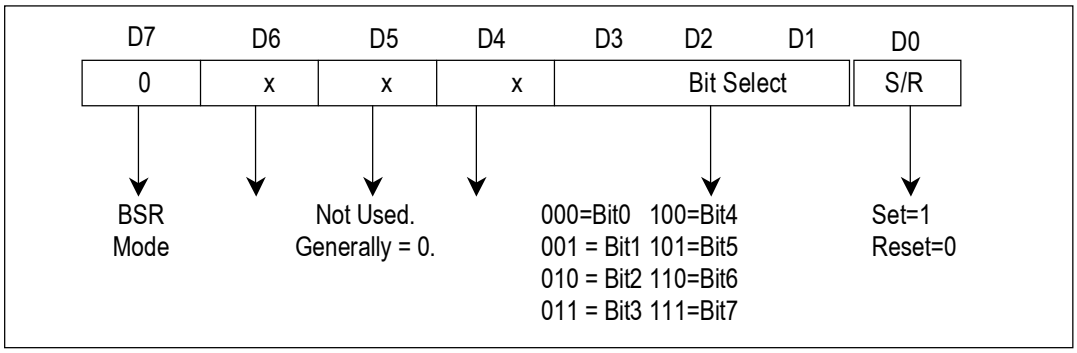


Hình 15.11: Nối ghép ADC 804 với 8255.

15.3 Các chế độ khác của 8255.

15.3.1 Chế độ thiết lập/ xoá bit BSR.

Một đặc tính duy nhất của cổng C là các bit có thể được điều khiển riêng rẽ. Chế độ BSR cho phép ta thiết lập các bit PC0 - PC7 lên cao xuống thấp như được chỉ ra trên hình 15.12. Ví dụ 15.6 và 15.7 trình bày cách sử dụng chế độ này như thế nào?

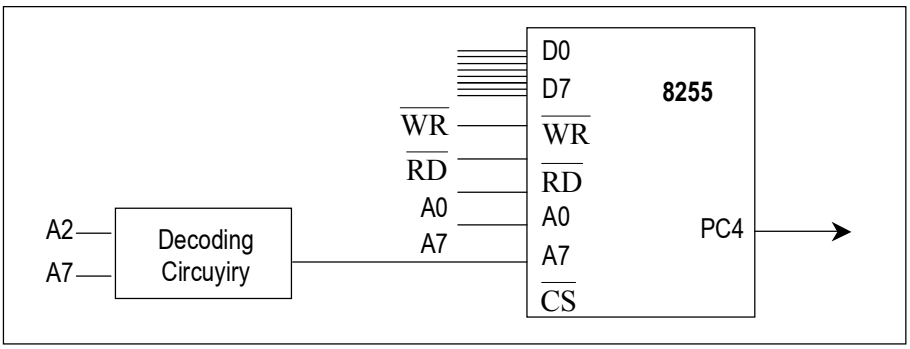


Hình 15.12: Từ điều khiển của chế độ BSR.

Ví dụ 15.6:

Hãy lập trình PCA của 8255 ở chế độ BSR thì bit D7 của từ điều khiển phải ở mức thấp. Để PC4 ở mức cao, ta cần một từ điều khiển là "0xxx1001" và ở mức thấp ta cần "0xxx1000". Các bít được đánh dấu x là ta không cần quan tâm và thường chúng được đặt về 0.

```
MOV     A, 00001001B    ; Đặt byte điều khiển cho PC4 =1
MOV     R1, #CNTPORT    ; Nạp cổng thanh ghi điều khiển
MOVX    @R1, A          ; Tạo PC4 = 1
ACALL   DELAY           ; Thời gian giữ chậm cho xung cao
MOV     A, #00001000B   ; Đặt byte điều khiển cho PC4 = 0
MOVX    @R1, A          ; Tạo PC4 = 0
ACALL   DELAY
```



Hình 15.13: Cấu hình cho ví dụ 15.6 và 15.7.

Ví dụ 15.7:

Hãy lập trình 8255 theo sơ đồ 15.13 để:

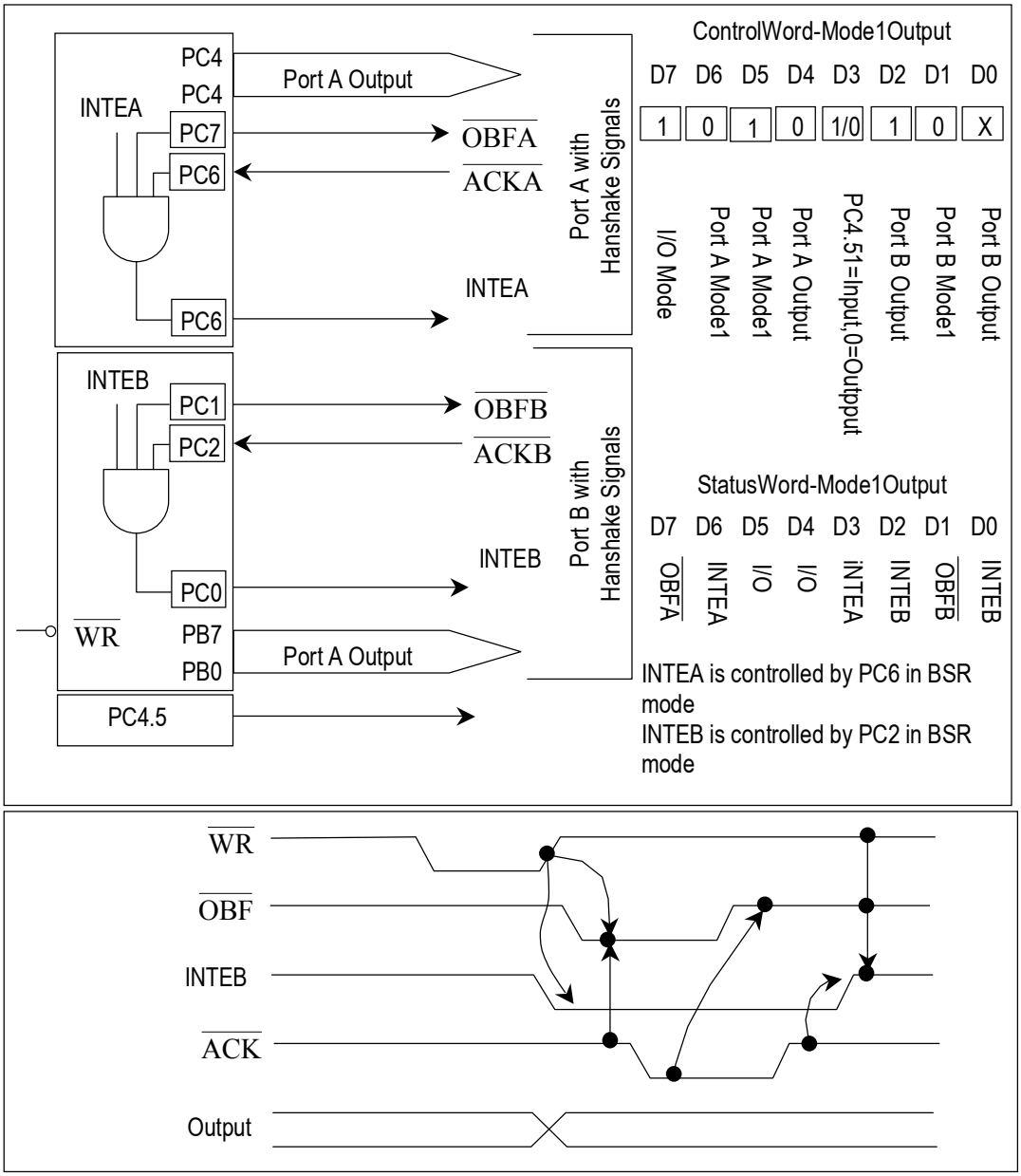
a) Đặt PC2 lên cao

b) Sử dụng PC6 để tạo xung vuông liên tục với 66% độ đầy xung.

Lời giải:

```
a) MOV R0, # CNTPORT
   MOV  A, # 0XXX0101                    ; Byte điều khiển
   MOV  @R0, A
b)
AGAIN:      MOV     A, #00001101B   ; Chọn PC6 = 1
            MOV     R0, #CNTPORT    ; Nạp địa chỉ thanh ghi điều khiển
            MOVX    @R0, A          ; Tạo PC6 = 1
            ACALL   DELAY
            ACALL   DELAY
            MOV     A, #00001100B   ; PC6 = 0
```

```
ACALL     DELAY  ; Thời gian giữ chậm
SJMP      AGAIN
```



Hình 15.15: Biểu đồ định thời của 8255 ở chế độ 1.

15.3.2 8255 ở chế độ 1: Vào/ ra với khả năng này bắt tay.

Một trong những đặc điểm mạnh nhất của 8255 là khả năng bắt tay với các thiết bị khác. Khả năng bắt tay là một quá trình truyền thông qua lại của hai thiết bị thông minh. Ví dụ về một thiết bị có các tín hiệu bắt tay là máy in. Dưới đây ta trình bày các tín hiệu bắt tay của 8255 với máy in.

Chế độ 1: Xuất dữ liệu ra với các tín hiệu bắt tay.

Như trình bày trên hình 15.14 thì cổng A và B có thể được sử dụng như các cổng đầu ra để gửi dữ liệu tới một thiết bị với các tín hiệu bắt tay. Các tín hiệu bắt tay cho cả hai cổng A và B được cấp bởi các bit của cổng C. Hình 15.15 biểu đồ định thời của 8255.

Dưới đây là các phần giải thích về các tín hiệu bắt tay và tính hợp lý của chúng đối với cổng A, còn cổng B thì hoàn toàn tương tự.

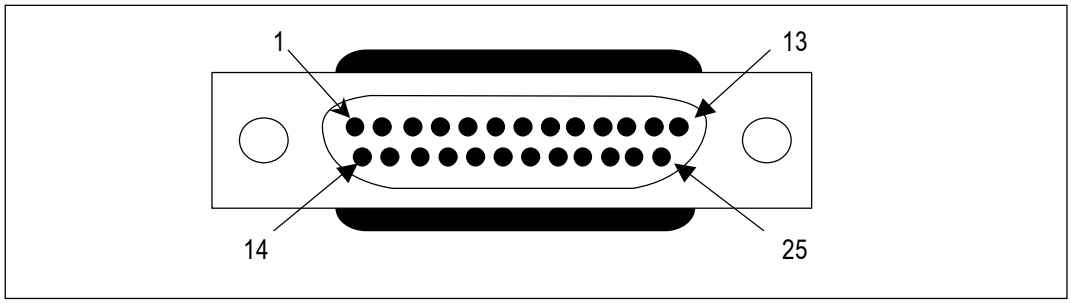
Tín hiệu $\overline{\text{OBFa}}$***:*** Đây là tín hiệu bộ đệm đầu ra đầy của cổng A được tích cực mức thấp đi ra từ chân PC7 để báo rằng CPU đã ghi 1 byte dữ liệu tới cổng A. Tín hiệu này phải được nối tới chân STROBE của thiết bị thu nhận dữ liệu (chẳng hạn như máy in) để báo rằng nó bây giờ đã có thể đọc một byte dữ liệu từ chốt cổng.

Tín hiệu $\overline{\text{ACKa}}$***:*** Đây là tín hiệu chấp nhận do cổng A có mức tích cực mức thấp được nhân tại chân PC6 của 8255. Qua tín hiệu $\overline{\text{ACKa}}$ thì 8255 biết rằng tín hiệu tại cổng A đã được thiết bị thu nhận lấy đi. Khi thiết bị nhận lấy dữ liệu đi từ cổng A nó báo 8255 qua tín hiệu $\overline{\text{ACKa}}$. Lúc này 8255 bật $\overline{\text{OBFa}}$ lên cao để báo rằng dữ liệu tại cổng A bây giờ là dữ liệu cũ và khi CPU đã ghi một byte dữ liệu mới tới cổng A thì $\overline{\text{OBFa}}$ lại xuống thấp v.v...

Tín hiệu INTRa: Đây là tín hiệu yêu cầu ngắt của cổng A có mức tích cực cao đi ra từ chân PC3 của 8255. Tín hiệu $\overline{\text{ACK}}$ là tín hiệu có độ dài hạn chế. Khi nó xuống thấp (tích cực) thì nó làm cho $\overline{\text{OBFa}}$ không tích cực, nó ở mức thấp một thời gian ngắn và sau đó trở nên cao (không tích cực). Sườn lên của $\overline{\text{ACK}}$ kích hoạt INTRa lên cao. Tín hiệu cao này trên chân INTRa có thể được dùng để gây chú ý của CPU. CPU được thông báo qua tín hiệu INTRa rằng máy in đã nhận byte cuối cùng và nó sẵn sàng để nhận byte dữ liệu khác. INTRa ngắt CPU ngừng mọi thứ đang làm và ép nó gửi byte kế tiếp tới cổng A để in. Điều quan trọng là chú ý rằng INTRa được bật lên 1 chỉ khi nếu INTRa, $\overline{\text{OBFa}}$ và $\overline{\text{ACKa}}$ đều ở mức cao. Nó được xoá về không khi CPU ghi một byte tới cổng A.

Tín hiệu INTEa: Đây là tín hiệu cho phép ngắt cổng A 8255 có thể cấm INTRa để ngăn nó không được ngắt CPU. Đây là chức năng của tín hiệu INTEa. Nó là một mạch lật Flip - Flop bên trong thiết kế để che (cấm) INTRa. Tín hiệu INTRa có thể được bật lên hoặc bị xoá qua cổng C trong chế độ BSR vì INTEa là Flip - Plop được điều khiển bởi PC6.

Từ trạng thái: 8255 cho phép hiển thị trạng thái của các tín hiệu INTR, OBF và INTE cho cả hai cổng A và B. Điều này được thực hiện bằng cách đọc cổng C vào thanh ghi tổng và kiểm tra các bit. Đặc điểm này cho phép thực thi thăm dò thay cho ngắt phần cứng.



Hình 15.16: Đầu cắm DB-25.

(hình 15.17 mờ quá không vẽ được)

Hình 15.17: Đầu cáp của máy in Centronics.

Bảng 15.2: Các chân tín hiệu của máy in Centronics.

Chân số	Mô tả	Chân số	Mô tả
1	STROBE	11	Bận (busy)
2	Dữ liệu D0	12	Hết giấy (out of paper)
3	Dữ liệu D1	13	Chọn (select)
4	Dữ liệu D2	14	Tự nạp ($\overline{\text{Autofeed}}$)
5	Dữ liệu D3	15	Lỗi ($\overline{\text{Error}}$)
6	Dữ liệu D4	16	Khởi tạo máy in
7	Dữ liệu D5	17	Chọn đầu vào ($\overline{\text{Select input}}$)
8	Dữ liệu D6	18-25	Đất (ground)
9	Dữ liệu D7		
10	$\overline{\text{ACK}}$ (chấp nhận)		

Các bước truyền thông có bắt tay giữa máy in và 8255.

Một byte dữ liệu được gửi đến bus dữ liệu máy in.

Máy in được báo có 1 byte dữ liệu cần được in bằng cách kích hoạt tín hiệu đầu vào STROBE của nó.

Khi máy nhận được dữ liệu nó báo bên gửi bằng cách kích hoạt tín hiệu đầu ra được gọi là $\overline{\text{ACK}}$ (chấp nhận).

Tín hiệu $\overline{\text{ACK}}$ khởi tạo quá trình cấp một byte khác đến máy in.

Như ta đã thấy từ các bước trên thì chỉ khi một byte dữ liệu tới máy in là không đủ. Máy in phải được thông báo về sự hiện diện của dữ liệu. Khi dữ liệu được gửi thì máy in có thể bận hoặc hết giấy, do vậy máy in phải được báo cho bên gửi khi nào nó nhận và lấy được dữ liệu của nó. Hình 15.16 và 15.17 trình các ổ cắm DB25 và đầu cáp của máy in Centronics tương ứng.