Experiment 5

Controlling DC Motors

Objective

The aim of this lab experiment is to control a small DC motor.

Equipment

Flight 8086 training board, Application board, PC with Flight86 software, download cable

Tasks to be Performed

- Running the motor in forward and reverse direction for a specified time
- Controlling the speed of the motor

5.1 DC Motor

The Application Board contains a small DC motor that can be operated in the forward or reverse direction. The operation of this DC motor is controlled by bits 6 and 7 on Port-A as shown in Table 5.1.

Table 5.1: Operation Modes of the DC Motor

Bit6	Bit7	Operation
0	0	Stop
0	1	Reverse Direction
1	0	Forward Direction
1	1	Stop

The following example shows you how to run the DC motor in the forward and reverse direction for a specific time.

Example 4.1: Write a program to run the DC motor in the forward direction for 5 seconds, turn it off for 3 seconds, then run it in the revere direction for 5 seconds.

Set SW2-1 to SWITCH
Set SW2-2 to MOTOR
SW4-1, SW4-2, SW4-3, and SW4-4 OFF

```
MOV   AL, 99h    ; initialize the 8255 PPI chip
OUT   06h, AL    ; A input, B output, C input
MOV   DL,    ?   ; load a proper value for 5s delay
MOV   AL,   40h  ; forward direction
OUT   02h, AL
CALL  Delay
MOV   DL,   ?    ; load a proper value for 3s delay
MOV   AL, 00h    ; stop the motor
OUT   02h, AL
CALL  Delay
MOV   DL,   ?    ; load a proper value for 5s delay
MOV   AL, 80h    ; reverse direction
OUT   02h, AL
CALL  Delay
MOV   AL, 00h    ; stop the motor
OUT   02h, AL
INT   5

; the delay procedure is left as an exercise
```

5.3 Controlling the Speed of the DC Motor

When the DC motor is ON (forward/reverse), it operates in its maximum speed. However, the speed of the motor can be controlled using *pulse width modulation* (PWM).

PWM is a common technique for speed control. A good analogy is bicycle riding. You peddle (exert energy) and then coast (relax) using your momentum to carry you forward. As you slow down (due to wind resistance, friction, road shape) you peddle to speed up and then coast again. The *duty cycle* is the ratio of peddling time to the total time (peddle+coast time). A 100% duty cycle means you are peddling all the time, and 50% only half the time.

PWM for motor speed control works in a very similar way. Instead of peddling, your motor is given a fixed voltage value (turned on) and starts spinning. The voltage is then removed (turned off) and the motor "coasts". By continuing this voltage on-off duty cycle, motor speed is controlled.

The concept of PWM inherently requires timing. The 8253 PIT chip can be used to generate PWM. In the beginning, the motor is turned on and Counter 0 is loaded with the ON duration. When Counter 0 terminates, the motor is turned off and Counter 1 is loaded with the OFF duration. Now, when Counter 1 terminates, the process is repeated from the beginning.

Example 4.2: Write a program to control the speed of the DC motor based on the state of Bit0 of the DIP switch. If Bit0 = 0, the motor will run at maximum speed. Otherwise, it will run at 50% of its duty cycle.

```
        Set SW2-1 to SWITCH
        Set SW2-2 to MOTOR
        SW4-1, SW4-2, SW4-3, and SW4-4  OFF
1       COMSEG SEGMENT BYTE PUBLIC 'CODE'
2       ASSUME CS:COMSEG, DS:COMSEG, ES:COMSEG, SS:COMSEG
3       ORG 0100h
4       Start:    ; set the external segment to point to the
5                 ; base of the Interrupt Vector Table (IVR)
6                 XOR AX,AX
7                 MOV ES,AX

8                 ;store the offset of ISR in the IVT
9                 MOV WORD PTR ES:[38*4],OFFSET IR6_ROUTINE
10                ;store the segment of ISR in the IVT
11                MOV WORD PTR ES:[38*4+2],CS

12                ;store the offset of ISR in the IVT
13                MOV WORD PTR ES:[39*4],OFFSET IR7_ROUTINE
14                ;store the segment of ISR in the IVT
15                MOV WORD PTR ES:[39*4+2],CS
```

```
16          ; initialize the 8255 PPI chip:
17          ; A and C input ports, B output port
18          MOV AL, 99h
19          OUT 06h, AL

20          ; initialize the 8259 PIC chip
21          MOV AL, 17h
22          OUT 10h, AL
23          MOV AL, 20h
24          OUT 12h, AL
25          MOV AL, 03h
26          OUT 12h, AL
27          MOV AL, 3Fh
28          OUT 12h, AL

29          ; initialize 8253 PIT chip  (00110000 = 30h)
30          ; Counter0, load MSB then LSB, mode 0, binary
31          MOV AL, 30h
32          OUT 0Eh, AL
33          ; initialize 8253 PIT chip  (01110000 = 70h)
34          ; Counter1, load MSB then LSB, mode 0, binary
35          MOV AL, 70h
36          OUT 0Eh, AL

37          ; counter0 loaded with FFFFh
38          MOV AL, 0FFh
39          OUT 08h, AL  ; first load low byte
40          MOV AL, 0FFh
41          OUT 08h, AL  ; now load high byte

42          STI          ; enable 8086 maskable interrupts

43          ; start of main program
44          MOV AL, 40h ; turn on the motor
45          OUT 02h, AL
46    again: JMP again   ; wait for interrupt on IR6/IR7
47                       ; Counter0/Counter1 decrements to 0

48    ; Interrupt Service Routine (ISR) for IR6
49    ; this routine checks Bit0 of the DIP switch
50    ; If Bit0 = 0 continue running the motor (max speed)
51    ; If Bit0 = 1 stop the motor (50% duty cycle)
52    ; the routine also reload Counter 1

53    IR6_ROUTINE:
54        IN   AL, 00h  ; read DIP switch

55        TEST AL, 01h  ; check Bit0
56        JZ   continue ; if Bit0=0 then don't stop the motor
57        MOV  AL, 00h  ; else stop the motor
58        OUT  02h, AL

59    continue:
60        ; counter1 loaded with FFFFh (50% duty cycle)
61        MOV AL, 0FFh
```

```
70        OUT 0Ah, AL   ; first load low byte
71        MOV AL, 0FFh
72        OUT 0Ah, AL   ; now load high byte
73        IRET

74    ; Interrupt Service Routine (ISR) for IR7
75    ; this routine turn on the motor and reload Counter 0

76    IR7_ROUTINE:
78
79              MOV AL,  40h  ; turn on the motor
80              OUT 02h, AL

81              ; counter0 loaded with FFFFh
82              MOV AL, 0FFh
83              OUT 08h, AL   ; first load low byte
84              MOV AL, 0FFh
85              OUT 08h, AL   ; now load high byte

86              IRET

87    COMSEG  ENDS
88    END     start
```

Exercises

5.1. Modify Example 5.2 to allow the user to control the direction in addition to the speed (Use Bit1 to control the direction).

5.2. Modify Example 5.2 to operate the motor at 4 different, for example 100% duty cycle, 50% duty cycle, 25% duty cycle, and 5 % duty cycle. The speed is selected based on the states of Bit0 and Bit1.