Faculty of Engineering
Electrical Engineering Department
Communications and Electronics Program

Student Lab Manual

CC 421

Microprocessors Lab

Lab Manual

Prepared by:

Prof. Dr. Mohamed El-Banna

Eng. Mostafa M. Ayesh Eng. Kareem Attaih

The lab exercises included in this manual are all base on the MTS-86C educational kits.

Index

CHAPTER 1:

INTERFACING THE 8255 – PARALLEL PORT CONTROL

1.1 OBJECTIVE

To utilize the 8255 programmable peripheral interface (PPI) to control LEDs.

1.2 DESCRIPTION

The aim of this lab is to use the 8086 microprocessor to control the 8255 PPI. This exercise shows how to use the PPI to produce certain patterns on a group of LEDs. In the first exercise, we demonstrate how to output a certain pattern permanently on the LED. The second exercise provides insights into how to create moving patterns and LED blinking. Before we go through exercises details, we explain how the 8255 interfacing works.

1.3 THE 8255 CHIP

The Intel 8255 PPI is a peripheral chip originally developed for the Intel 8085 microprocessor and was later used with other types (e.g. the 8086). The 8255 is used to give the CPU access to programmable parallel I/O.

1.3.1 Functionality

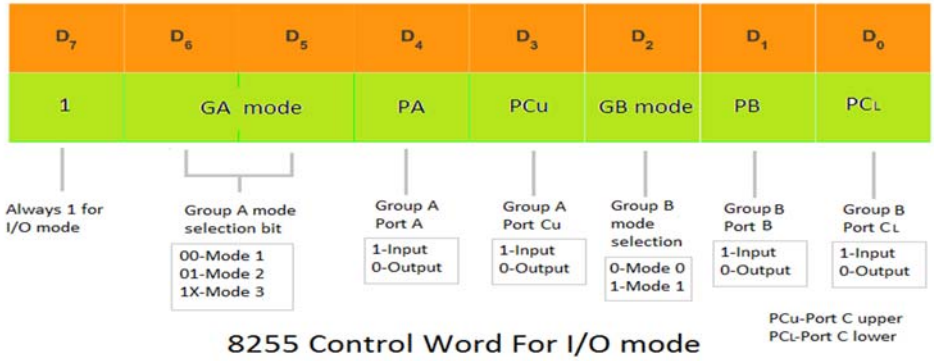
The 8255 has 24 input/output pins in total. These are divided into three 8-bit ports (portA, portB, and portC). portA and portB can be used as 8-bit I/O ports. portC can be used as an 8-bit I/O port or as two 4-bit I/O ports or even to provide handshaking signals for ports A and B.

The three groups are further grouped as follows:

1- Group A (GA) consisting of portA and the upper part of port C.
2- Group B (GB) consisting of portB and the lower part of port C.

1.3.2 The Control register

The address line pins provided on chip allow access to any of the previously mentioned ports or the control register; an 8-bit register which controls the operating nature of each port on chip. The control word format is given below.



8255 Control Word For I/O mode

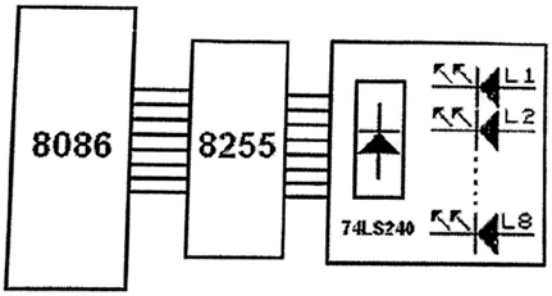
1.3.3 Group modes

As can be seen from the above diagram, the control register allows different functionality modes for GA and GB by manipulating the D_6, D_5, and D_2. Modes discussions and description can be obtained by referring to original chip data sheet. In this lab, we are only concerned with mode 1 operation for GA and mode 0 for GB which provides the following functionalities:

1. Output ports are latched.
2. Input ports are buffered, not latched.
3. Ports do not handshake.

1.4 SIMPLIFIED SCHEMATIC

The 8086 microprocessor controls the 8255 chips which interfaces the LED group connected on portB. The 8 bits LEDs are driven by 74LS240. If some pin on portB is logic 1, the signal will be reversed by the 74LS240 and the corresponding LED will turn on. On the other hand, if that pin is logic 0. The corresponding LED will turn off.


8086
8255
74LS240
L1
L2
L8

Experiment #1

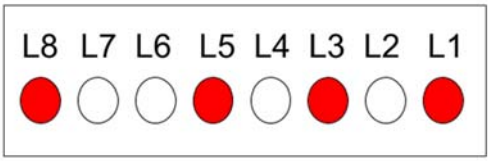
Output 95H to 8 bit LEDS

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

1. EXERCISE (1): OUTPUT 95H TO 8 BIT LEDS

1.1 OBJECTIVE

When the program is executed the, the following pattern (95H) will be shown on the LEDS permanently.



1.2 SOURCE PROGRAM

```
=================================== 8255-1.ASM ================================

CNT3            EQU   3FD6H          ; Define 8255 control word port address
BPORT3          EQU   3FD2H          ; Define 8255 portB address

CODE            SEGMENT
                ASSUME  CS:CODE, DS:CODE

                ORG 0


; <Setup 8255 control word register>

START:          MOV   SP, 4000H      ; Setup stack pointer
                MOV   AL, 90H
                MOV   DX, CNT3       ; Enable 8255 control word
                OUT   DX, AL         ; Output data 90H to 8255 control port

; <Output 95H to LED>
```

```
                MOV     AL, 95H
                MOV     DX, BPORT3              ; Enable 8255 portB
                OUT     DX, AL                  ; Output data 95H to portB

CODE            ENDS
                END     START
```

========== 8255-1.ASM ==========

1.3 PROGRAM DESCRIPTION

- First the Addresses of the control register (3FD6H) and port B (3FD2H) are assigned to CNT3, and BPORT3.
- Next a byte of value 90H is written to the control register of the 8255. This is equivalent to: activating the chip for I/O operations ($D_7 = 1$), portA is set to input port ($D_4 = 1$), portB and portC are set to output ports ($D_3 = 0$, $D_1 = 0, D_0 = 0$) and finally modes of operations for the two groups are chosen as mode 0 ($D_6 = 0$, $D_5 = 0, D_2 = 0$).
- A value of 95H is sent to portB to light the LEDs according to the desired sequence.

1.4 TASKS

- **Task (1):** Modify the above source code to output 7AH.
- **Task (2):** Using the source code obtained in task (1), try to write a code that alternately displays 95H, 7AH on the LED group (Note: you may need to set a delay between the shown patterns so that your eyes are able to capture the change in the displayed sequences).

Experiment #2

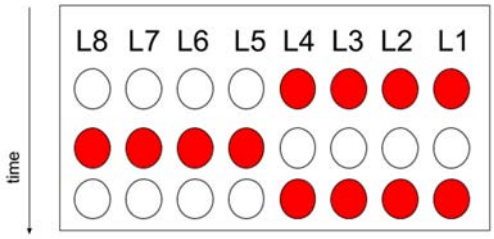
LED Blinking Control

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

2. EXERCISE (2): LED BLINKING CONTROL

2.1 OBJECTIVE

When executed, the program will have the following effect on the LEDs. These patterns will be displayed repeatedly on the LEDS.



2.2 Source Program

```
======================================== 8255-2.ASM ==================================

CNT3            EQU     3FD6H                   ; Define 8255 control word port address
BPORT3          EQU     3FD2H                   ; Define 8255 portB address

CODE            SEGMENT
                ASSUME  CS:CODE, DS:CODE

                ORG 0


; <Setup 8255 control word register>

START:          MOV     SP, 4000H               ; Setup stack pointer
                MOV     AL, 90H
                MOV     DX, CNT3                ; Enable 8255 control word
                OUT     DX, AL                  ; Output data 90H to 8255 control port

; <Output 0FH and F0H to LEDs repeatedly>
```

```
                MOV     AL, 0FH             ; Initialize AL with 0FH
                MOV     DX, BPORT3          ; Enable 8255 portB
J1:             OUT     DX, AL              ; Output data 0FH to portB
                MOV     CX, 0A000H          ; Set CX as a counter
                LOOP    $                   ; A Loop that does nothing (time delay)
                NOT     AL                  ; Invert the date in AL
                JMP     J1                  ; Repeat

CODE            ENDS
                END     START
```

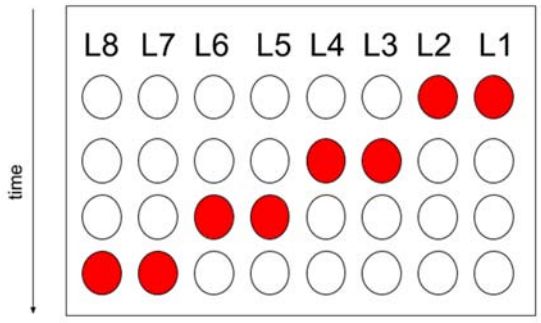
========== 8255-2.ASM ==========

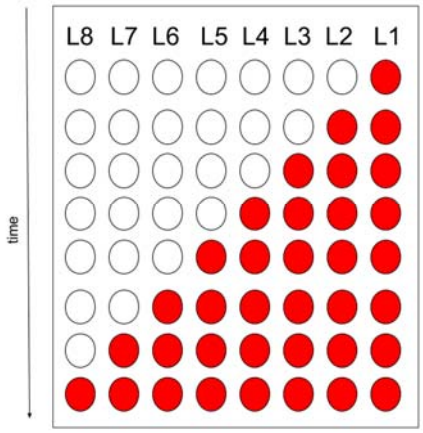
2.3 Program description

- After initializing the control word, a value of 0FH is sent to portB.
- A delay is introduces via the LOOP instruction.
- The LOOP instruction decreases CX every time the loop command is executed. LOOP terminates and control is passed to the next instruction when CX is zero. The loop command is simply a command that does nothing.
- Next the data in AL is inverted and the process is repeated to give rise to the desired patterns.

2.4 Tasks

- **Task (3):** Write a source code to produce the following patterns. These patterns are to be shown repeatedly. You may use the rotate left command (ROL BX, 1) which rotates register BX to the left.





CHAPTER 2:

DIGITAL TO ANALOG CONVERTER

2.1 OBJECTIVE

To utilize the DAC on chip to generate different waveforms

2.2 DESCRIPTION

The digital to analog converter (DAC or D/A Converter) is used in transforming the digital signals to analog signals for control, information display, or further analog processing. The objective of this unit is to program DAC0808 chip – this chip is on kit connected with 8086 microprocessor – it will be programmed by different techniques to generate different waveforms.

Experiment #3

Sawtooth wave generation

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

3. EXERCISE (3): SAWTOOTH WAVE GENERATION

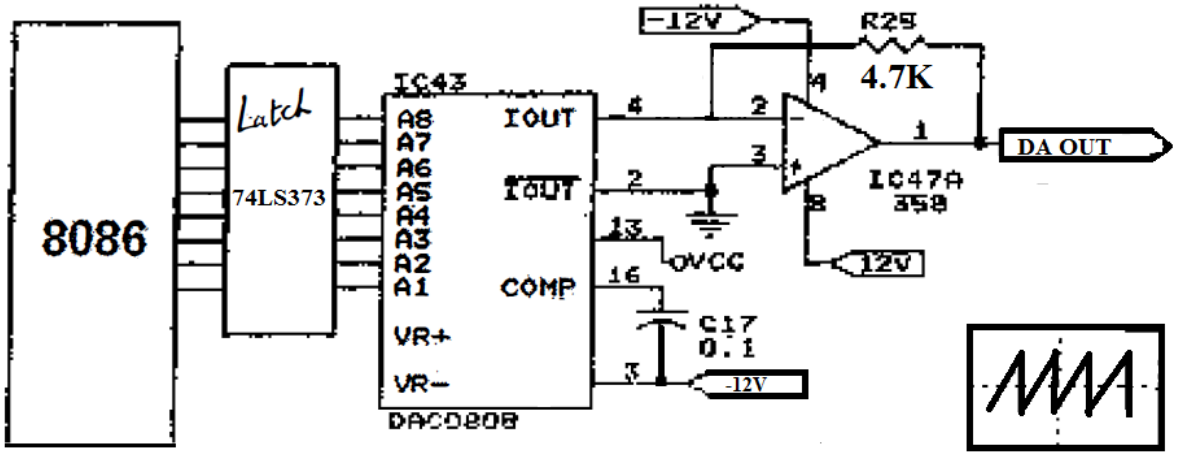
3.1 Objective

When the program is executed, the sawtooth waveform is generated at DAC0808 output.

3.2 Circuit Description

The output signal programmed from **8086** is latched by **74LS373** and send to **DAC0808** chip.

These signals are programmed from register **AL** and each input to **DAC0808** produces a single and discrete value at **DA OUT** (named **TP1** at MTS-86C kit panel).



Functional circuit description of exercise (1)

3.3 Source Program

```
======================================== DAC-1.ASM ==================================

DAC             EQU     3FD8H                   ; Define DAC0808 port address

CODE            SEGMENT
                ASSUME  CS:CODE, DS:CODE

                ORG 0
```

```
START:      XOR   AL,AL          ; initialize AL=0
            MOV   DX, DAC        ; Enable DAC0808

; <Generating sawtooth wave>

J1:         OUT   DX, AL         ; Output data to DAC0808
            INC   AL             ; AL+1
            JMP   J1             ; Jump to J1

CODE        ENDS
            END   START
```

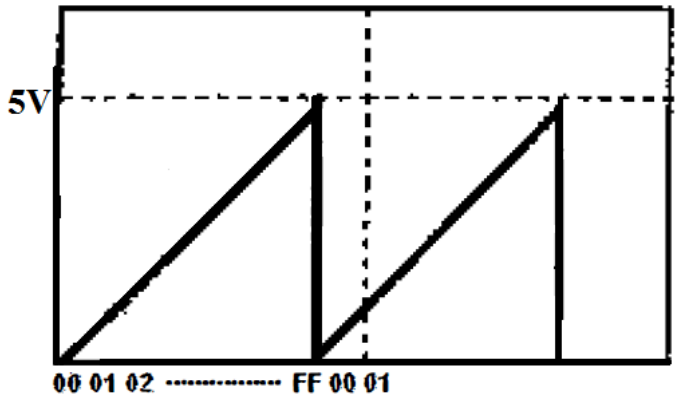
========== DAC-1.ASM ==========

3.4 Program Description

- The resolution of DAC is defined as the smallest difference of the analog output when the digital input changes a unit count. The input scales range from **00H-FFH** (256 scales), and the reference voltages is 5V in the system. As the results the resolution of the output is 5V/256 = 0.01953125V.
- In this program, each output loop increase AL by 1, meaning that adding 0.01953125V to the output

3.5 Results

- Use oscilloscope to observe the output at **TP1**. The result is a sawtooth waveform, as shown in the figure below



Result on oscilloscope after executing source program

3.6 Tasks

- **Task (1):** Modify the above source code to output a **sawtooth** biased by **0.625V** and its peak-peak voltage is **1.25V**.

Experiment #4

Triangular wave generation

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

4. EXERCISE (4): TRIANGULAR WAVE GENERATION

4.1 Objective

When the program is executed, the triangular wave is generated at DAC0808 output.

4.2 Circuit description

Same as previous exercise

4.3 Source Program

```
====================================== DAC-2.ASM ==================================

DAC             EQU     3FD8H                   ; Define DAC0808 port address

CODE            SEGMENT
                ASSUME  CS:CODE, DS:CODE

                ORG 0

START:          XOR     AL,AL                   ; initialize AL=0
                MOV     DX, DAC                 ; Enable DAC0808

; <Generating triangular wave rising routine>

J1:             OUT     DX, AL                  ; Output data to DAC0808
                INC     AL                      ; AL+1

                CMP AL,0FFH                     ; Maximum value (FFH)?

                JNZ     J1                      ; If not, Jump to J1

; <Generating triangular wave falling routine>

J2:             OUT     DX, AL                  ; Output data to DAC0808
                DEC     AL                      ; AL-1

                AND     AL,AL                   ; Minimum Value (00H)?

                JNZ     J2                      ; If not, Jump to J2

                JMP     J1                      : Jump to J1
```

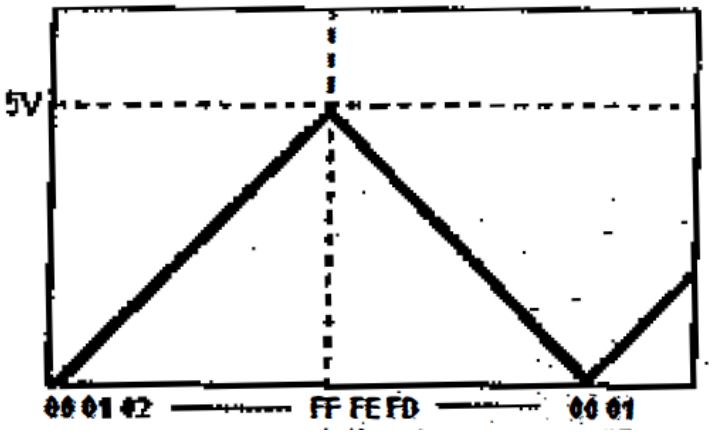
```
CODE                    ENDS
                        END     START
======================================== DAC-2.ASM ==================================
```

4.4 Program description

- Loop J1 output the signal until it reaches maximum counts (FFH) and loop J2 output the signal until it reaches minimum counts (OOH).

4.5 Results

- Use oscilloscope to observe the output at **TP1**. The result is a triangular waveform, as shown in the figure below



Result on oscilloscope showing triangular waveform

4.6 Tasks

- **Task (2):** modify the source code of triangular waveform to be biased by **1.25V** and its peak to peak wave form is **2.5V**
- **Task (3):** write a code to generate 25% duty cycle rectangular wave and what is the name of 50% duty cycle rectangular wave?

CHAPTER 3:

ANALOG TO DIGITAL CONVERSION

1.5 OBJECTIVE

To utilize the ADC chip in order to convert analog signals from different sensors to digital signal and display on LED and PC.

1.6 DESCRIPTION

The analog to digital converter translates from analog measurements, which are usually continuous voltages and currents, to digital words used in computing; data transmission, information processing and storage. The objective of this unit is to program the ADC0809 chip to convert different sensing signal to signal and display the results on LED and PC.

Three exercises are available in this unit:

Exercise (5): Variable resistor
Exercise (6): Phototransistor
Exercise (7): Thremistor

Experiment #5

Variable Resistor

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

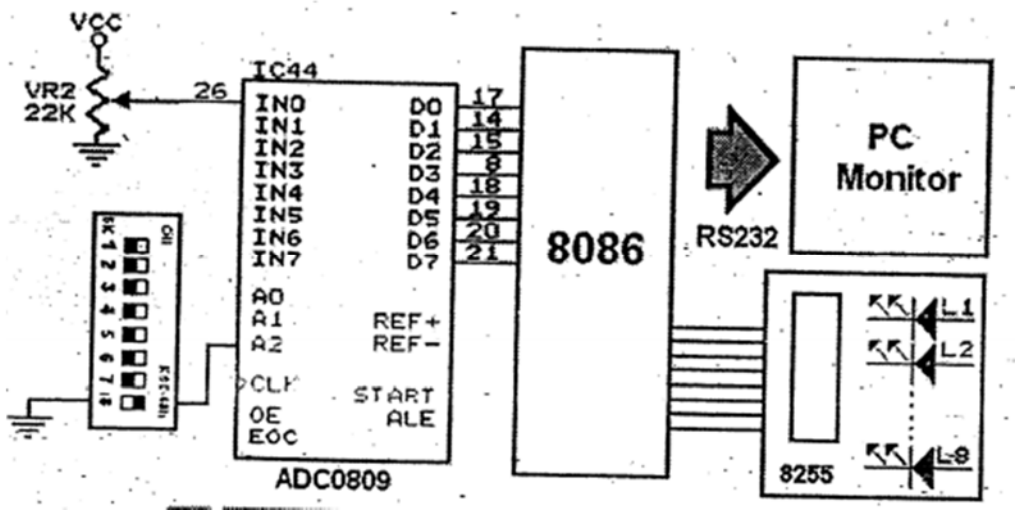
5. EXERCISE (5): VARIABLE RESISTOR

5.1 Objective

When the program is executed, the voltage input from the variable resistor circuit is converted to binary format on LED and the value is displayed on PC.

5.1.1 Circuit Description

The voltage applied to IN0 of ADC0809 is connected to variable resistor VR2. This voltage is converted to 8 bit digital code and sent to LED with binary format and to PC with decimal format. To enable ADC0809, bit 8 (D/A) of DIPSW3 should be at “on” state.



5.1.2 Source Program

========== ADC-1.ASM==========

```
ADC0        EQU     3FC8H
CNT3        EQU     3FD6H
BPORT3      EQU     3FD2H


CODE        SEGMENT
            ASSUME          CS:CODE, DS:CODE

            ORG     0
```

```
START:  MOV   SP,4000H
        MOV   AX,CS
        MOV   DS,AX

        MOV   DX,CNT3
        MOV   AL,91H
        OUT   DX,AL


        MOV   DX,OFFSET MSG
        MOV   AH,9
        INT   21H

J1:     MOV   AX,0
        MOV   DX,ADC0
        OUT   DX,AL
        MOV   CX,20H
        LOOP  $


        IN    AL,DX
        MOV   DX,BPORT3
        OUT   DX,AL


        MOV   DX,OFFSET DATA1
        MOV   AH,9
        INT   21H


        MOV   DH,2
        INT   10H


        MOV   DX,OFFSET DATA2
        MOV   AH,9
        INT   21H
```

```
            CALL    CONVERT
            MOV     CX,0
            LOOP    $
            JMP     J1


CONVERT:    MOV     AH,0
            MOV     BL,51
            DIV     BL

            MOV     DL,AL
            OR      DL,30H


            MOV     AL,AH
            MOV     AH,2
            INT     21H

            MOV     DL,'.'
            MOV     AH,2
            INT     21H

            MOV     BL,10
            MUL     BL
            MOV     BL,51
            DIV     BL

            MOV     DL,AL
            OR      DL,30H
            MOV     AL,AH
            MOV     AH,2
            INT     21H

            MOV     BL,10
            MUL     BL
            MOV     BL,51
            DIV     BL

            MOV     DL,AL
            OR      DL,30H
```

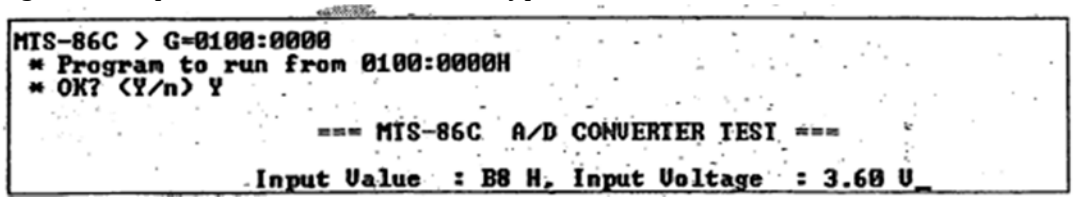
```
                MOV     AL,AH
                MOV     AH,2
                INT     21H

                MOV     DL,' '
                INT     21H
                MOV     DL,'V'
                INT     21H
                RET


MSG             DB      0DH,0AH,9,9,'    === MTS-86C  A/D CONVERTER TEST ==='
                DB      0DH,0AH,0AH,9,9,9,'$'

DATA1:          DB      0DH,9,9,'Input Value  : $'

DATA2:          DB      ' H, Input Voltage  : $'


CODE            ENDS

                END     START
```

5.1.3 Program Description

The resolution of the ADC0809 is 8 bits. The analog input is divided into 2^8 or 256 discrete ranges. With 5v reference voltage, each range represents 5V/256 = 0.01953V. Thus the digital output code 00H corresponds to an analog input voltage of 0.00V, and FFH represents 5V.

5.1.4 Results

When adjusting the variable resistor VR2, the converted digital output will show at 8 bit LED and the voltage value input to ADC will show at PC hyper terminal, as shown below.



Note: Make sure that bit 8 of DIPSW3 is ON.

Experiment #6

PHOTOTRANSISTOR

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

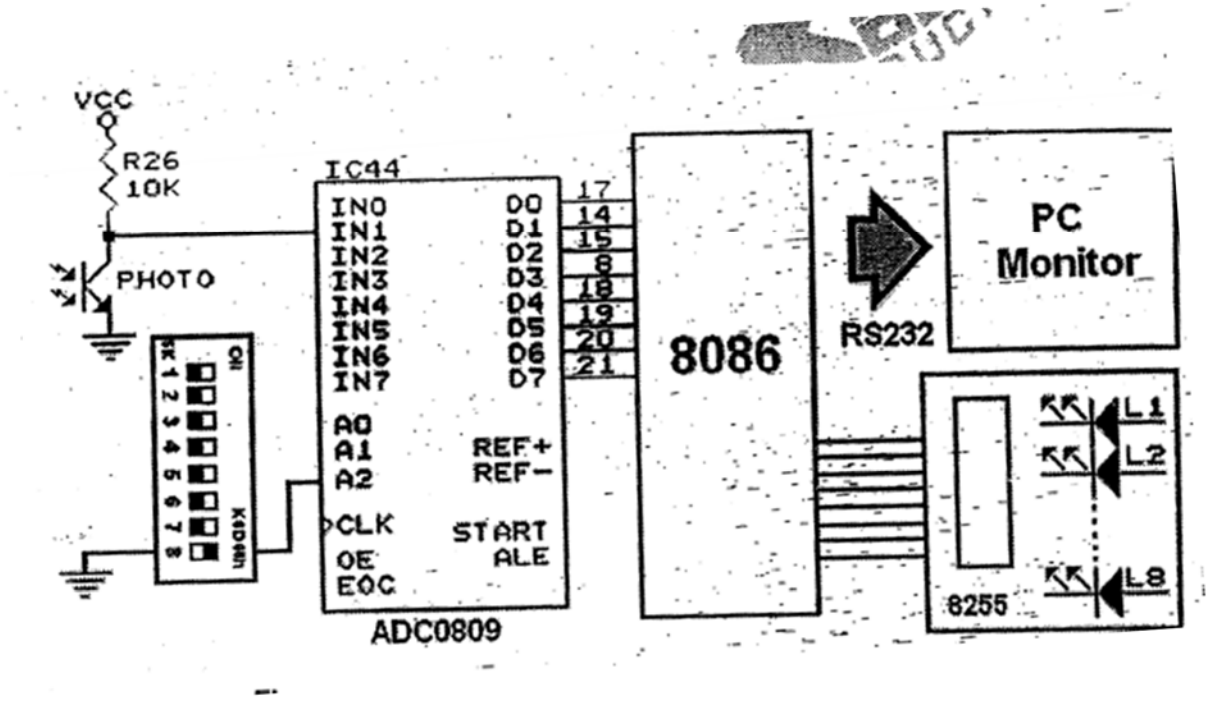
6. Exercise (6): **PHOTOTRANSISTOR**

6.1.1 Objective

When the program is executed, the voltage input from phototransistor sensing circuit is converted to binary format on LED and hexadecimal format on PC.

6.1.2 Circuit Description

The phototransistor is used in a reverse-bias state in collector-base junction with open base. The junction photocurrent increases with the increasing incident light. As a result, the current flowing through the phototransistor is proportional to the light injected to the base of the device, making lower voltage drop at collector. In other words, the voltage applied to IN1 of ADC0809 in inverse proportional to the magnitude of the light injected to the phototransistor.



6.1.3 Source Program

========== ADC-2.ASM ==========

```
ADC1        EQU   3FCAH
CNT3        EQU   3FD6H
BPORT3      EQU   3FD2H

CODE        SEGMENT
```

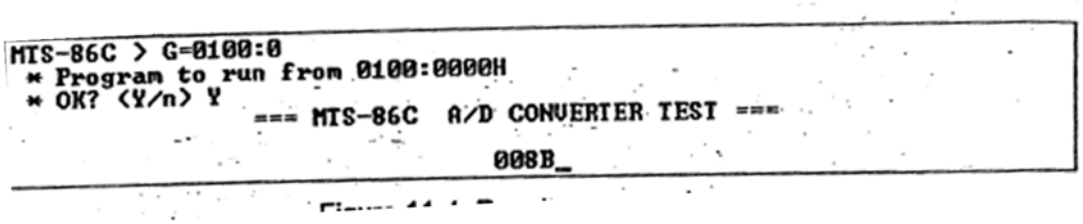
```
                ASSUME      CS:CODE, DS:CODE

                ORG     0


START:          MOV     SP,4000H
                MOV     AX,CS
                MOV     DS,AX

                MOV     DX,CNT3
                MOV     AL,91H
                OUT     DX,AL

                MOV     DX,OFFSET MSG
                MOV     AH,9
                INT     21H

J1:             MOV     AX,0
                MOV     DX,ADC1
                OUT     DX,AL


                MOV     CX,20H
                LOOP    $

                IN      AL,DX
                MOV     DX,BPORT3
                OUT     DX,AL

                MOV     DH,4
                INT     10H

                MOV     CX,0
                LOOP    $

                MOV     DX,OFFSET BACK
                MOV     AH,9
                INT     21H

                JMP     J1
```

```
MSG     DB    9,9,'=== MTS-86C  A/D CONVERTER TEST ==='
        DB    0DH,0AH,0AH,9,9,9,9,'$'
BACK    DB    8,8,8,8,'$'

CODE    ENDS
        END   START
```

6.1.4 Results:

When injecting the light to the phototransistor, the binary valued output at LED and hexadecimal valued output at PC hyper terminal decreases and vice verse.

The output at PC hyper terminal is shown below.



Note: Make sure that bit 8 of DIPSW3 is ON.

Experiment #7

THERMISTOR

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

7. Exercise (7): THERMISTOR

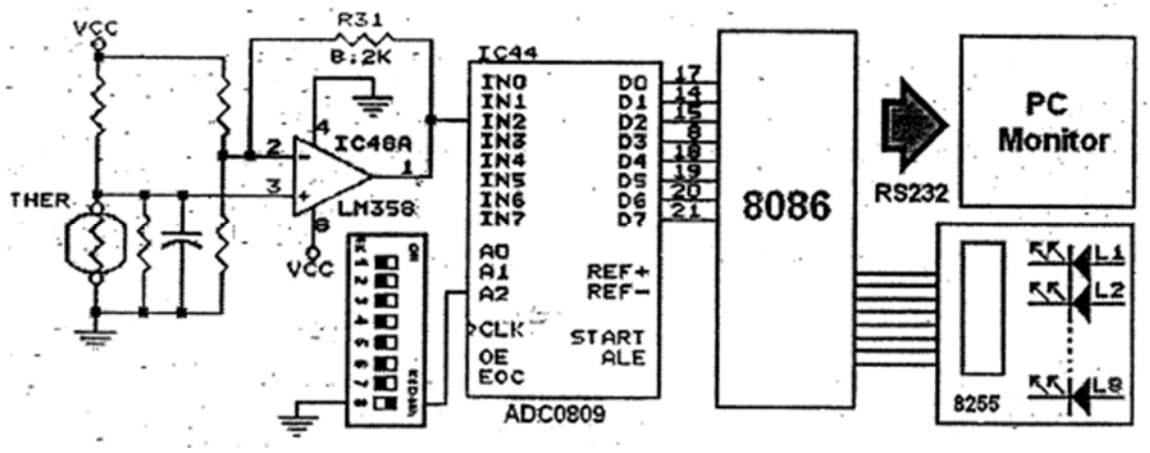
7.1.1 Objective

When the program is executed, the voltage input from the Thermistor sensing circuit is converted to binary format on LED and hexadecimal format on PC.

7.1.2 Circuit Description

Thermistor (an abbreviation for thermal-sensitive resistors) is characterized by its small size, fast time constant, high negative temperature coefficient, and wide range of available base resistance. The output of the Thermistor is connected to an amplification circuit. This is because the current following through the Thermistor must be kept very low to assure near zero power dissipation and near-zero self-heating.

The type of Thermistor used in MTS-86C is a negative temperature coefficient. In other words, the voltage applied to IN2 of ADC0809 is inversely proportional to the heat absorbed by the Thermistor.



7.1.3 Source Program

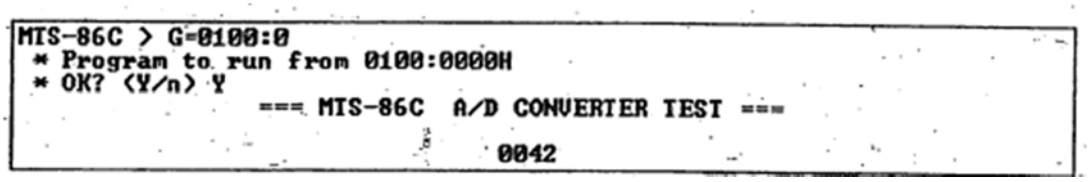
========== ADC-3.ASM ==========

```
ADC2        EQU   3FCCH
CNT3        EQU   3FD6H
BPORT3      EQU   3FD2H

CODE        SEGMENT

            ASSUME        CS:CODE, DS:CODE

            ORG   0

START:      MOV   SP,4000H
            MOV   AX,CS
            MOV   DS,AX

            MOV   DX,CNT3
            MOV   AL,91H
            OUT   DX,AL



            MOV   DX,OFFSETMSG
            MOV   AH,9
            INT   21H
```

```
J1:     MOV   AX,0
        MOV   DX,ADC2
        OUT   DX,AL

        MOV   CX,20H
        LOOP  $

        IN    AL,DX

        MOV   DX,BPORT3
        OUT   DX,AL

        MOV   DH,4
        INT   10H

        MOV   CX,0
        LOOP  $

        MOV   DX,OFFSET   BACK
        MOV   AH,9
        INT   21H

        JMP   J1

MSG     DB    9,9,'===  MTS-86C  A/D  CONVERTER  TEST  ==='
        DB    0DH,0AH,0AH,9,9,9,9,'$'

BACK    DB    8,8,8,8,'$'

CODE    ENDS

        END   START
```

7.1.4 Results

When heating the Thermistor, the binary valued output at LED and hexadecimal valued output at PC hyperterminal decreases and vice versa.



CHAPTER 4:

PROGRAMMABLE TIMER / COUNTER CONTROL – 8253

4.1 OBJECTIVE

To utilize the 8253 on chip to create computer music.

4.2 DESCRIPTION

8253 is a programmable timer/counter designed for timing applications control. The technique used in this unit involves programming the timer to generate the square waves at various audio frequencies and send these waveforms to the speaker.
Detailed circuit schematic diagram of each exercise is shown in its section and detailed operation of 8253 chip is provided in this link

Experiment #8

Music performed by 8253

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

8. EXERCISE (8): MUSIC PERFORMED BY 8253

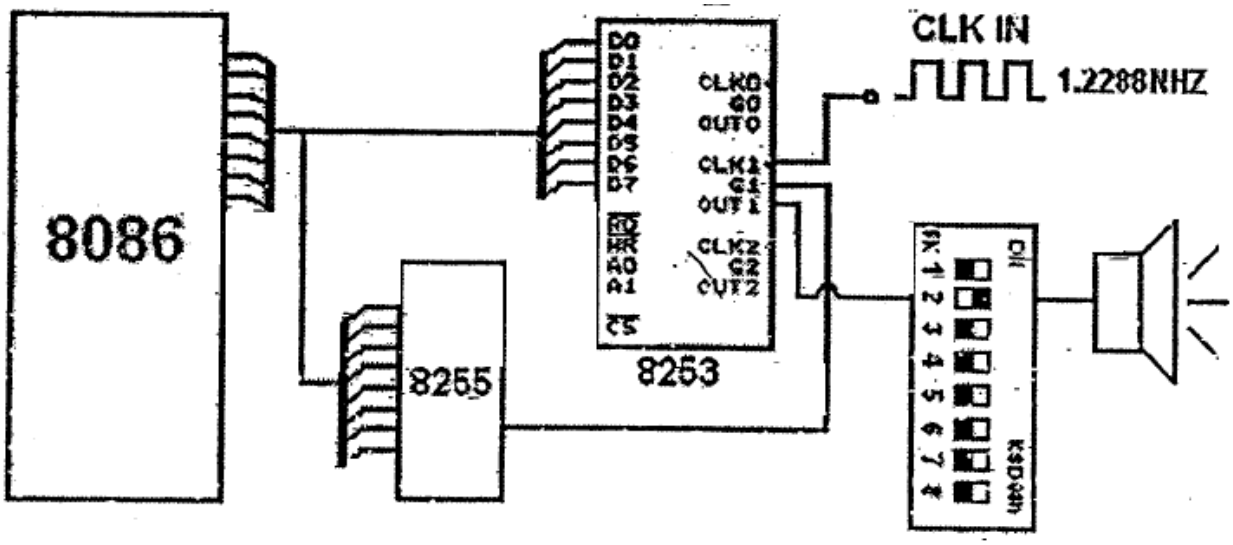
8.1 Objective

When the program is executed, the speaker will play according to the following music note.

Scale	$\overline{1}$	$\overline{1}$	$\overline{1}$	6	5	6	5	3	3	–
Beat	(4)	(4)	(4)	(2)	(2)	(2)	(2)	(4)	(4)	(4)
Scale	6	6	6	5	3	5	3	2	2	–
Beat	(4)	(4)	(4)	(2)	(2)	(2)	(2)	(4)	(4)	(4)
Scale	$\overline{1}$	$\overline{1}$	$\overline{1}$	6	5	6	5	3	3	–
Beat	(4)	(4)	(4)	(2)	(2)	(2)	(2)	(4)	(4)	(4)
Scale	2	6	5	3	5	6	6	6	–	–
Beat	(4)	(4)	(4)	(2)	(2)	(4)	(4)	(4)	(4)	

8.2 Circuit Description

The speaker is driven by an operational amplifier from the square wave output of **8253** counter **#1**, which is synchronized by a **1.2288MHz** clock. The output waveform and output frequency are controlled by **8253** control word register and count register. The **bit4** of **8255 portC** is used to control the gating between **8253** and speaker through **DIPSW3 bit 2**, as the result, the second switch of **DIPSW3** must be turned ON.



Functional circuit description of exercise (1)

8.3 Source Program

========== 8253-1.ASM ==========

```
COUNT1          EQU   0FFDAH
CSR       EQU   0FFDEH
CNT3            EQU        3FD6H
CPORT3          EQU     3FD4H

DO        EQU   0
RE        EQU   2
MI        EQU   4
FA        EQU   6
SOL       EQU   8
RA        EQU   0AH
SY        EQU   0CH
DO1       EQU   0EH
NO        EQU   10H

          ORG   0

CODE      SEGMENT
          ASSUME CS:CODE,DS:CODE

START:    MOV   SP,4000H
          MOV   AX,CS
          MOV   DS,AX
          MOV   ES,AX

          MOV   DX,CNT3
          MOV   AL,80H
          OUT   DX,AL

          MOV   DX,CPORT3
          MOV   AL,0FFH
          OUT   DX,AL

          MOV   DX,OFFSET MSG
          MOV   AH,9
          INT   21H

I8253:    MOV   DX,CSR
          MOV   AL,01110110B
          OUT   DX,AL
```

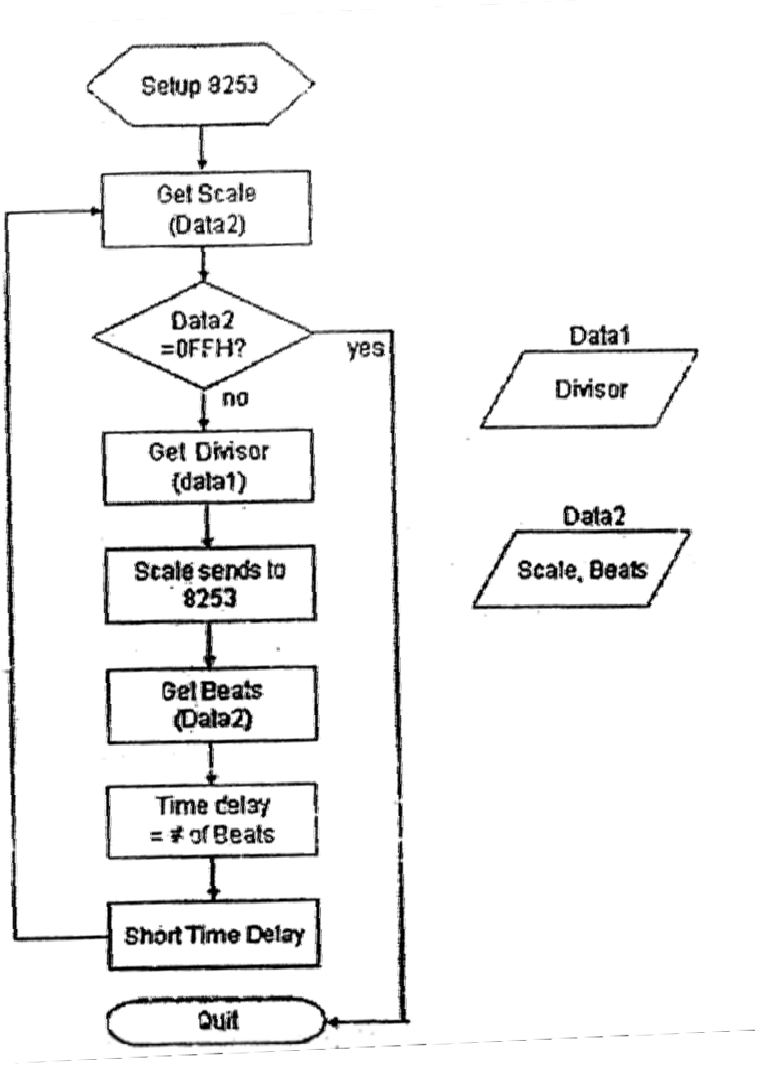
```
            MOV   DX,COUNT1

PLAY:       MOV   BX,OFFSET DATA1
            MOV   SI,OFFSET DATA2
MAIN:       MOV   AL,[SI]
            CMP   AL,0FFH
            JZ    SONG_END
            MOV   AH,0
            PUSH  SI
            MOV   SI,AX
            MOV   AX,[BX+SI]
            OUT   DX,AL
            MOV   AL,AH
            OUT   DX,AL
            POP   SI
            INC   SI
            MOV   AL,[SI]

J1:         CALL  TIMER
            DEC   AL
            JNZ   J1
                  MOV   AX,10
            OUT   DX,AL
            MOV   AL,AH
            OUT   DX,AL
            MOV   CX,0AFH
            LOOP  $
            INC   SI
            JMP   MAIN

SONG_END:   MOV   AH,4CH
            INT   21H

TIMER:            MOV   CX,2800H
            LOOP  $
            RET

DATA1:            ;          DO,   RE,   MI,   FA
            DW    4697,4184,3728,3519
            ;         SOL,   RA,   SY, DO1, NO
            DW    3135,2793,2491,2352,10H

DATA2:            DB
      DO1,4,DO1,4,DO1,4,RA,2,SOL,2,RA,2,SOL,2,MI,4,MI,4,NO,4
            DB    RA,4,RA,4,RA,4,SOL,2,MI,2,SOL,2,MI,2,RE,4,RE,4,NO,4
            DB    DO1,4,DO1,4,DO1,4,RA,2,SOL,2,RA,2,SOL,2,MI,4,MI,4,NO,4
            DB    RE,4,RA,4,SOL,4,MI,2,SOL,2,RA,4,RA,4,RA,4,NO,4
```

```
            DB    0FFH

MSG         DB    0DH,0AH,0AH
            DB    9,9,9,'============================',0DH,0AH
            DB    9,9,9,'=== SPEAKER TEST PROGRAM ===',0DH,0AH
            DB    9,9,9,'===    MTS-86C (8253)    ===',0DH,0AH
            DB    9,9,9,'============================',0DH,0AH,0AH
            DB    9,9,'      Select the DIP Switch3 (OUT1-
SP)',0DH,0AH,'$'

CODE        ENDS
            END   START
```

========== 8253-1.ASM ==========

8.4 Program Description

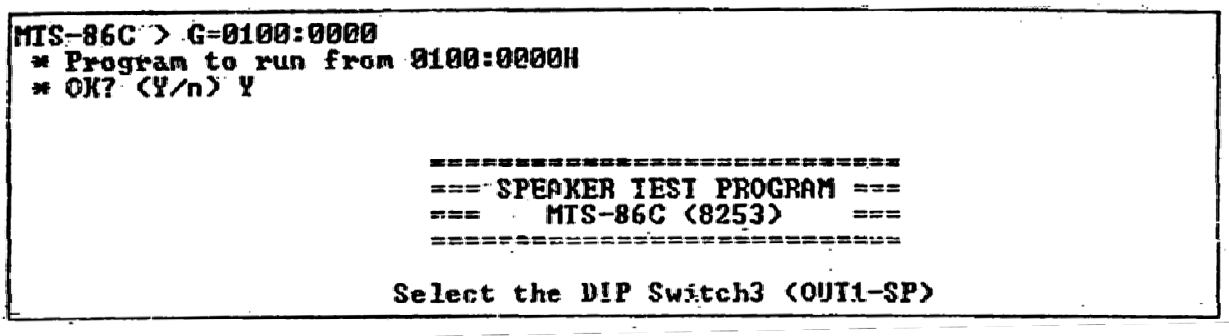
- A simple control flow chart of 8253-1.ASM is shown below. Basically, the scale and beats inside data2 are read alternately. When the scale is read, program lookup the divisor from data1 and send the count number to 8253. The beat time is stored in beat data in data2 and controlled by the time delay routine. Which defines the time for unit beat



Simple control flowchart of 8253-1.ASM

8.5 Results

- After the program is downloaded to MTS-86C, it can be executed wither from Hyper Terminal software or from the MTS-86C. when the program is executed from Hyper Terminal software, the window will display the title “SPEAKER TEST PROGRAM” and then perform the music from MTS-86C speaker, as shown below



- When the program is executed from MTS-86C, it will perform the music directly from MTs-86C speaker. Using debug function will help to understand the code flow of the program.
- Note:
- Be sure that bit 2 of DIPSW3 is ON

Experiment #9

Piano- input from keyboard

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

9. EXERCISE (9): PIANO- INPUT FROM KEYBOARD

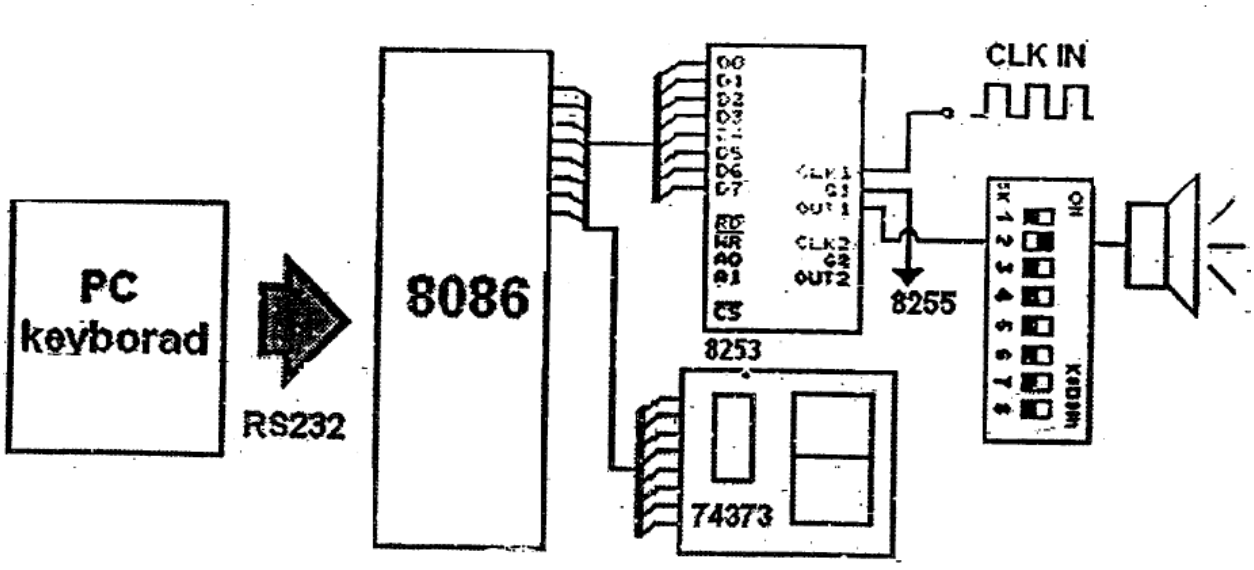
9.1 Objective

When the program is executed, the signal input from PC keyboard will show at the FND, and will convert to corresponding music scale. The music scale will be performed on MTS-86C speaker. The relation between input key, FND and music scale is shown in below table.

Key	Z / z	X / x	C / c	V / v	B / b	N / n	M / m	,
FND	7	6	5	4	3	2	1	0
Musical Scale	DO	RE	MI	FA	SOL	RA	SY	DO1

9.2 Circuit description

The input signal from PC keyboard sends to 8086 through RS-232 cable and output to speaker and FND.



9.3 Source Program

========== 8253-3.ASM ==========

```
COUNT1          EQU     0FFDAH

CNT3            EQU     3FD6H

CPORT3          EQU     3FD4H
```

```
CSR             EQU     0FFDEH
FND             EQU     3FF0H

                ORG     0

CODE    SEGMENT
        ASSUME CS:CODE,DS:CODE

START:          MOV     SP,4000H
                MOV     AX,CS
                MOV     DS,AX

                MOV     DX,CNT3
                MOV     AL,90H
                OUT     DX,AL

                MOV     DX,CPORT3
                MOV     AL,0FFH
                OUT     DX,AL

I8253:  MOV     DX,CSR
        MOV     AL,01110110B
        OUT     DX,AL
```

```
        MOV     DX,COUNT1

PLAY:   MOV     AH,0
        INT     21H

        CALL    NOSOUND
        PUSH    DX

PLAY1:  MOV     DX,FND

        CMP     AL,'Z'
        JZ      DO
        CMP     AL,'z'
        JZ      DO

        CMP     AL,'X'
        JZ      RE
        CMP     AL,'x'
        JZ      RE

        CMP     AL,'C'
        JZ      MI
        CMP     AL,'c'
```

```
JZ      MI

CMP     AL,'V'
JZ      FA
CMP     AL,'v'
JZ      FA

CMP     AL,'B'
JZ      SOL
CMP     AL,'b'
JZ      SOL

CMP     AL,'N'
JZ      RA
CMP     AL,'n'
JZ      RA

CMP     AL,'M'
JZ      SY
CMP     AL,'m'
JZ      SY

CMP     AL,','
```

```
        JZ      DO1
        JMP     PAUSE

DO:     MOV     AL,11011000B  ;7
        OUT     DX,AL
        MOV     AX,4697               ;DO
        JMP     SET8253

RE:     MOV     AL,10000010B  ;6
        OUT     DX,AL
        MOV     AX,4184               ;RE
        JMP     SET8253

MI:     MOV     AL,10010010B  ;5
        OUT     DX,AL
        MOV     AX,3728               ;MI
        JMP     SET8253

FA:     MOV     AL,10011001B  ;4
        OUT     DX,AL
        MOV     AX,3519               ;FA
        JMP     SET8253
```

```
SOL:        MOV   AL,10110000B  ;3
            OUT   DX,AL
            MOV   AX,3135             ;SOL
            JMP   SET8253

RA:         MOV   AL,10100100B  ;2
            OUT   DX,AL
            MOV   AX,2793             ;RA
            JMP   SET8253

SY:         MOV   AL,11111001B  ;1
            OUT   DX,AL
            MOV   AX,2491             ;SI
            JMP   SET8253

DO1:        MOV   AL,11000000B  ;0
            OUT   DX,AL
            MOV   AX,2352             ;DO
            JMP   SET8253

PAUSE:      MOV   AX,10

SET8253:    POP   DX
```

```
                OUT     DX,AL
        MOV     AL,AH
        OUT     DX,AL
                JMP     PLAY

NOSOUND:        PUSH    AX
                MOV     AX,10
                OUT     DX,AL
                MOV     AL,AH
                OUT     DX,AL
                MOV     CX,1500
                LOOP    $
                POP     AX
                RET

CODE            ENDS
                END     START
```

========== 8253-3.ASM ==========

9.4 Program description

- The programming flow is using a technique called “polling”, this method is used to detect which switch is pressed from MTS-86C keypad or PC keyboard. In this exercise we will use a software interrupt method to detect which key is pressed from PC keyboard.

9.5 Results

When z or Z is pressed, FND shows 7, and speaker performs “DO”.

When x or X is pressed, FND shows 6, and speaker performs “RE”.

When c or C is pressed, FND shows 5, and speaker performs “MI”.

When v or V is pressed, FND shows 4, and speaker performs “FA”.

When b or B is pressed, FND shows 3, and speaker performs “SOL”.

When n or N is pressed, FND shows 2, and speaker performs “RA”.

When m or M is pressed, FND shows 1, and speaker performs “SY”.

When comma “,” is pressed, FND shows 0, and speaker performs “DO1”.

Note: Be sure that bit 2 of DIPSW3 is ON.

Experiment #10

Music performed by 8255

Preparation	In Class Activity	Results	Total	Signature
20%	40%	40%	100%	

10. EXERCISE (10): MUSIC PERFORMED BY 8255

10.1 Objective

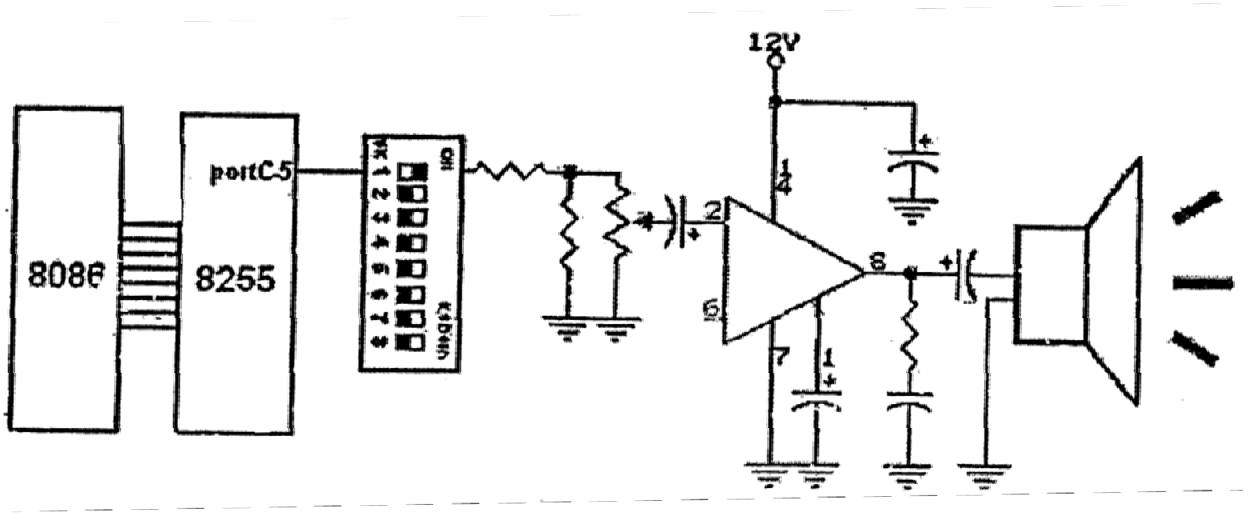
When the program is executed, the speaker will play according to the following music note.

Scale	$\overline{1}$	$\overline{1}$	$\overline{1}$	6	5	6	5	3	3	–
Beat	(4)	(4)	(4)	(2)	(2)	(2)	(2)	(4)	(4)	(4)
Scale	6	6	6	5	3	5	3	2	2	–
Beat	(4)	(4)	(4)	(2)	(2)	(2)	(2)	(4)	(4)	(4)
Scale	$\overline{1}$	$\overline{1}$	$\overline{1}$	6	5	6	5	3	3	–
Beat	(4)	(4)	(4)	(2)	(2)	(2)	(2)	(4)	(4)	(4)
Scale	2	6	5	3	5	6	6	6	–	–
Beat	(4)	(4)	(4)	(2)	(2)	(4)	(4)	(4)	(4)	

10.2 Circuit Description

The speaker is driven by an operational amplifier from the high/low output of **8255** portC-**5**.

The output frequency and duration is controlled by 8066 registers. The bit5 of 8255 portC is used to connect to the amplifier through bit 1 of DIPSW3, as the result, the first switch of DIPSW3 must be turned ON.



10.3 Source Program

========== 8253-4.ASM ==========

```
CNT3            EQU     3FD6H
```

```
CPORT3      EQU   3FD4H

CODE        SEGMENT
            ASSUME      CS:CODE, DS:CODE

            ORG   0

START:      MOV   SP,8000
            MOV   AX,CS
            MOV   DS,AX

            MOV   AL,80H
            MOV   DX,CNT3
            OUT   DX,AL


            MOV   BX,0F00H
            MOV   SI,4000
      MOV   [SI],BX

      MOV   BX,OFFSET DATA1
            CALL  PLAY
            MOV   CX,9FFFH
```

```
        LOOP    $
    MOV BX,OFFSET DATA2
        CALL    PLAY
        MOV     CX,9FFFH
        LOOP    $
    MOV BX,OFFSET DATA3
        CALL    PLAY
        MOV     CX,9FFFH
        LOOP    $
    MOV BX,OFFSET DATA4
        CALL    PLAY
        MOV     AH,4CH
        INT     21H


PLAY:   MOV     CL,[BX]
        INC     BX
        MOV     CH,[BX]
        MOV     AL,CH
        AND     AL,AL
        JZ      RETURN
        PUSH    BX
        MOV     AL,0FFH
```

```
                MOV     DX,CPORT3
        CALL    COUNT
                POP     BX
        INC     BX
                MOV     CX,0FFFH
                LOOP    $
        JMP     PLAY

RETURN:         RET

COUNT:          MOV     BX,[SI]
J3:             NOT     AL
                OUT     DX,AL
                MOV     AH,CL

J1:             DEC     BX
                CMP     BH,0
                JZ      J2
        DEC     AH
        JNZ     J1
                JMP     J3
J2:             DEC     CH
                JNZ     COUNT
```

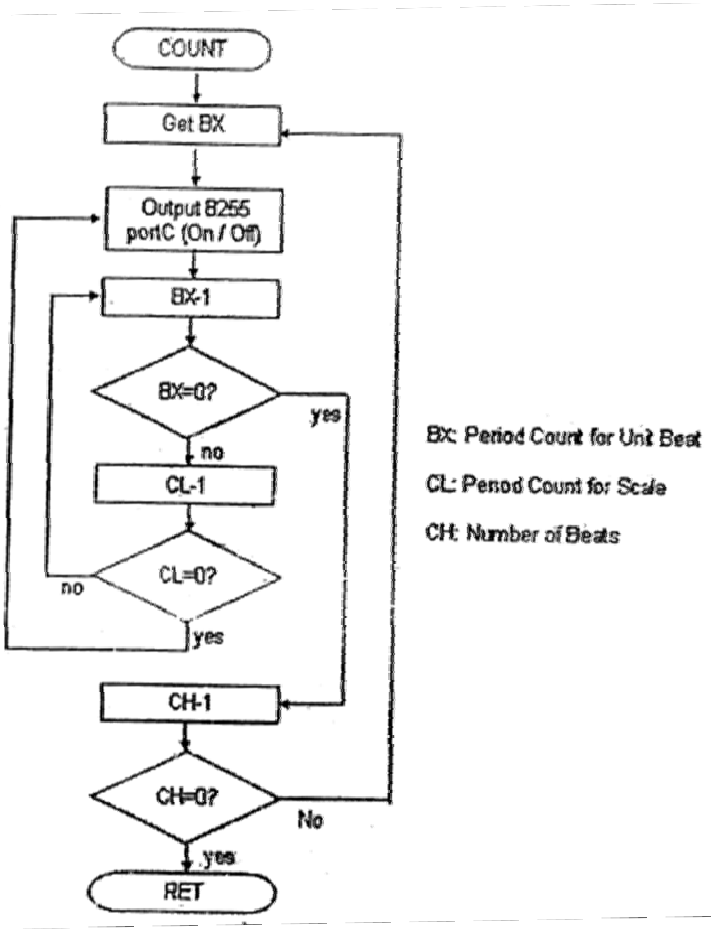

```
                RET

DATA1:          DB
        19H,04H,19H,04H,19H,04H,1EH,02H,22H,02H,1EH,02H,22H,02H,28H,04H,28H,04H,00H,00H

DATA2:          DB
        1EH,04H,1EH,04H,1EH,04H,22H,02H,28H,02H,22H,02H,28H,02H,2DH,04H,2DH,04H,00H,00H

DATA3:          DB
        19H,04H,19H,04H,19H,04H,1EH,02H,22H,02H,1EH,02H,22H,02H,28H,04H,28H,04H,00H,00H

DATA4:          DB
        2DH,04H,1EH,04H,22H,04H,28H,02H,22H,02H,1EH,04H,1EH,04H,1EH,5H,00H,00H

CODE            ENDS

                END     START
====================================== 8253-4.ASM ==================================
```

10.4 Program description

The sound is generated by counting the value of 8086 registers to control 8255 portC-5 output (high/low) duration.

The scale and beats of the sound stored in the sound table is read alternately and save to CL and CH respectively.

Next figure illustrates the flowchart of COUNT routine. Refer to code will help you to understand the counting register usage in the flowchart.



10.5 Results

When the program is executed from MTS-86C, it will perform the music directly from MTs-86C speaker.