



Electrical and Computer Systems Engineering Department

Computer Design Laboratory

ENCS411

Report 2

Experiment 7: Generating Music using 8254 PIT on PC

Prepared by

Anas Nimer_1180180

Instructor:

Dr. Abdallatif Abuissa

Teacher assistant:

Eng. Motasem Diab

BIRZEIT

August – 2020

1. Abstract

In this experiment, we will understand, configure and test the 8253/4 Programmable Interval Timer (PIT) devices and apply some examples using DOSBOX tool on the personal computer.

Table of Contents

1. Abstract.....	I
2. Theory	1
2.1. 8255A - Programmable Peripheral Interface PPI.....	1
2.1.1. Ports of 8255A	1
2.1.2. Operating Modes.....	2
2.1.3. Features of 8255A.....	2
2.1.4. Data Bus Buffer	2
2.1.5. Read/Write Control Logic.....	2
2.1.6. CS	3
2.1.7. WR.....	3
2.1.8. RESET	3
2.1.9. RD.....	3
2.1.10. A0 and A1	3
2.2. 8253 - Programmable Interval Timer PIT.....	4
2.2.1. Features of 8253 / 54	4
2.2.2. Data Bus Buffer	4
2.2.3. Read/Write Logic	5
2.2.4. Control Word Register	5
2.2.5. Counters.....	6
3. Procedure & Discussion.....	7
3.1 Task 1	7
3.2. Part A:- Generating Beep using debug	8
3.2.1. PartA.6.1 : Frequency = 1 KHZ.....	9
3.2.2. Part A.6.2 : Frequency = 5 KHZ.....	9
3.2.3. Part A.6.2 : Frequency = 15 KHZ.....	10
3.3. Part B:- Using TASM to produce beep sounds	10
3.3.1. Task B.3:	11
3.3.2. Task B.4:	11
3.4. Part C:- Generate Music on your PC.....	12
3.5. Part D:- Using Keyboard as Piano keys.....	13
4. Conclusion	14
5. References.....	15
6. Appendix.....	16
6.1. A.1.....	16

6.2.	A.2.....	17
6.3.	A.3.....	18
6.4.	B.1.....	20
6.5.	C.1.....	23

Table of Table:

Table 1: (ports addresses)	3
Table 2:(Counters).....	5
Table 3: (Addresses For Ports)	7

Table of Contents

Figure 1 (8254 PIT).....	4
Figure 2 (Control Register Configuration).....	5
Figure 3: (Part A.1).....	8
Figure 4 (F out = 1 KHz)	9
Figure 5 (F out = 5 KHz)	9
Figure 6 (F out = 15 KHz)	10
Figure 7 (Task B.3 Tasm & Tlink)	11
Figure 8 (Some Piano Notes And Frequencies)	12
Figure 9 (Part C Tasm & Tlink).....	12
Figure 10 (Part D Tasm & Tlink).....	13

2. Theory

In the PC there is a single clock used to synchronize activities of all peripheral chips connected to the CPU. The clock, which has the highest frequency in the system, belongs to the CPU. There are functions within the PC that require a clock with a lower frequency. The PIT (8253/54) is used to bring down the frequency to the desired level for various uses such as the beep sound in the PC. The 8254 PIT provides three independent channel timers that are programmed using the control (command) register of the PIT.[1]

2.1. 8255A - Programmable Peripheral Interface PPI

The 8255A is a general-purpose programmable I/O interface that can pass data from I/O to interrupt I/O under defined conditions. It is compatible with virtually all microprocessors.

It consists of three 8-bit bidirectional I/O ports (24I/O lines) which can be configured as per the requirement.

2.1.1. Ports of 8255A

8255A has three ports, i.e., PORT A, PORT B, and PORT C.

- ✓ Port A contains one 8-bit output latch/buffer and one 8-bit input buffer.
- ✓ Port B is similar to PORT A.
- ✓ Port C can be split into two parts, i.e. PORT C lower (PC0-PC3) and PORT C upper (PC7-PC4) by the control word.

These three ports are further divided into two groups, i.e. Group A includes PORT A and upper PORT C. Group B includes PORT B and lower PORT C. These two groups can be programmed in

three different modes, i.e. the first mode is named as mode 0, the second mode is named as Mode 1 and the third mode is named as Mode 2.

2.1.2. Operating Modes

8255A has three different operating modes –

- Mode 0 – In this mode, Port A and B is used as two 8-bit ports and Port C as two 4-bit ports. Each port can be programmed in either input mode or output mode where outputs are latched and inputs are not latched. Ports do not have interrupt capability.

- Mode 1 – In this mode, Port A and B is used as 8-bit I/O ports. They can be configured as either input or output ports. Each port uses three lines from port C as handshake signals. Inputs and outputs are latched.

- Mode 2 – In this mode, Port A can be configured as the bidirectional port and Port B either in Mode 0 or Mode 1. Port A uses five signals from Port C as handshake signals for data transfer. The remaining three signals from Port C can be used either as simple I/O or as handshake for port B.

2.1.3. Features of 8255A

The prominent features of 8255A are as follows –

- It consists of 3 8-bit IO ports i.e. PA, PB, and PC.
- Address/data bus must be externally demux'd.
- It is TTL compatible.
- It has improved DC driving capability.

2.1.4. Data Bus Buffer

It is a tri-state 8-bit buffer, which is used to interface the microprocessor to the system data bus. Data is transmitted or received by the buffer as per the instructions by the CPU. Control words and status information is also transferred using this bus.

2.1.5. Read/Write Control Logic

This block is responsible for controlling the internal/external transfer of data/control/status word. It accepts the input from the CPU address and control buses, and in turn issues command to both the control groups.

2.1.6. CS

It stands for Chip Select. A LOW on this input selects the chip and enables the communication between the 8255A and the CPU. It is connected to the decoded address, and A0 & A1 are connected to the microprocessor address lines.

Their result depends on the following conditions :-

CS	A1	A0	Result
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	Control Register
1	X	X	No Selection

Table 1: (ports addresses)

2.1.7. WR

It stands for write. This control signal enables the write operation. When this signal goes low, the microprocessor writes into a selected I/O port or control register.

2.1.8. RESET

This is an active high signal. It clears the control register and sets all ports in the input mode.

2.1.9. RD

It stands for Read. This control signal enables the Read operation. When the signal is low, the microprocessor reads the data from the selected I/O port of the 8255.

2.1.10. A0 and A1

These input signals work with RD, WR, and one of the control signal.

2.2. 8253 - Programmable Interval Timer PIT

The Intel 8253 and 8254 are Programmable Interval Timers (PTIs) designed for microprocessors to perform timing and counting functions using three 16-bit registers. Each counter has 2 input pins, i.e. Clock & Gate, and 1 pin for “OUT” output. To operate a counter, a 16-bit count is loaded in its register. On command, it begins to decrement the count until it reaches 0, then it generates a pulse that can be used to interrupt the CPU.

2.2.1. Features of 8253 / 54

The most prominent features of 8253/54 are as follows :

- It has three independent 16-bit down counters.
- It can handle inputs from DC to 10 MHz.
- These three counters can be programmed for either binary or BCD count.
- It is compatible with almost all microprocessors.
- 8254 has a powerful command called READ BACK command, which allows the user to check the count value, the programmed mode, the current mode, and the current status of the counter.

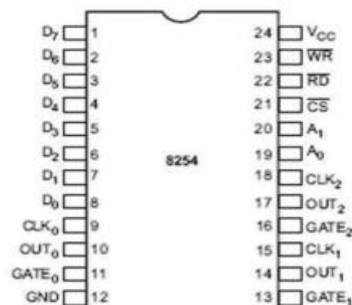


Figure 1 (8254 PIT)

In the above figure, there are three counters, a data bus buffer, Read/Write control logic, and a control register. Each counter has two input signals - CLOCK & GATE, and one output signal – OUT .

2.2.2. Data Bus Buffer

It is a tri-state, bi-directional, 8-bit buffer, which is used to interface the 8253/54 to the system data bus. It has three basic functions :-

- ✓ Programming the modes of 8253/54.
- ✓ Loading the count registers.
- ✓ Reading the count values.

2.2.3. Read/Write Logic

It includes 5 signals, i.e. RD, WR, CS, and the address lines A0 & A1. In the peripheral I/O mode, the RD and WR signals are connected to IOR and IOW, respectively. In the memory mapped I/O mode, these are connected to MEMR and MEMW. Address lines A0 & A1 of the CPU are connected to lines A0 and A1 of the 8253/54, and CS is tied to a decoded address. The control word register and counters are selected according to the signals on lines A0 & A1.

A1	A0	Result
0	0	Counter 0
0	1	Counter 1
1	0	Counter 2
1	1	Control Word Register
X	X	No Selection

Table 2:(Counters)

2.2.4. Control Word Register

This register is accessed when lines A0 & A1 are at logic 1. It is used to write a command word, which specifies the counter to be used, its mode, and either a read or write operation.



Figure 2 (Control Register Configuration)

2.2.5. Counters

Each counter consists of a single, 16 bit-down counter, which can be operated in either binary or BCD. Its input and output is configured by the selection of modes stored in the control word register. The programmer can read the contents of any of the three counters without disturbing the actual count in process.

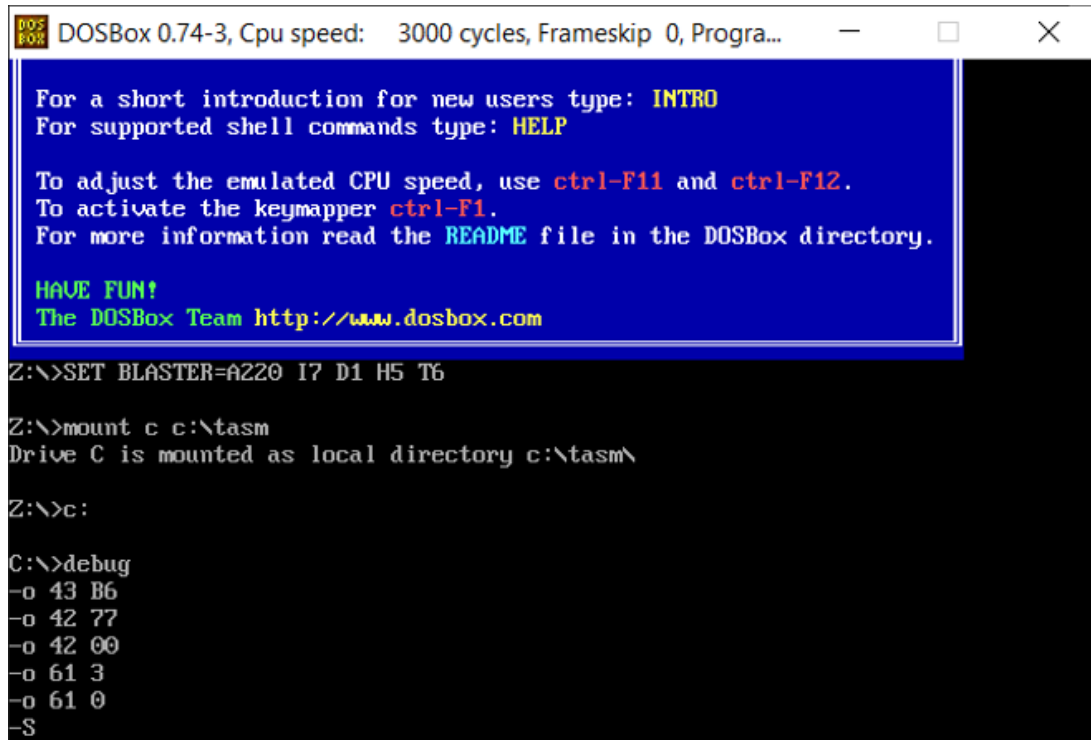
3. Procedure & Discussion

3.1. Task 1

Port	Address in PC
PPI Port A	60 H
PPI Port B	61 H
PPI Port C	62 H
PPI Command register	63 H
PIT Counter 0	40 H
PIT Counter 1	41 H
PIT Counter 2	42 H
PIT Command register	43 H

Table 3: (Addresses For Ports)

3.2. Part A:- Generating Beep using debug



```
DOS FOR DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount c c:\tasm
Drive C is mounted as local directory c:\tasm\

Z:\>c:

C:\>debug
-o 43 B6
-o 42 77
-o 42 00
-o 61 3
-o 61 0
-S
```

Figure 3: (Part A.1)

To use counter two, read and write LSB first, mode 3 and binary counter the control register will be set to 10110110, which is B6 in Hexadecimal.

In the second line Counter 2 value assigned to counter 2 address which is 42H where its value can be found through the following equation

$$\text{counter 2} = F_{\text{in}} / F_{\text{out}}$$

clk2 is connected with 1.19 MHz which is the input frequency where the output 10 KHz then :-

$$\text{counter 2} = F_{\text{in}} / F_{\text{out}} = 1.19 \text{ MHz} / 10 \text{ KHz} = 0077 \text{ h.}$$

it's assigned least significant first (77 H) then max significant (00) to counter 2 address (42 H) then to run speaker PB0 & PB1 data assigned to 11 to Port B (61 H) in the end 00 assigned to 61H to turn off the speaker .

3.2.1. PartA.6.1 : Frequency = 1 KHZ

counter 2 = $F_{in} / F_{out} = 1.19 \text{ MHz} / 1 \text{ KHz} = 4A6h$.

```
Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount c c:\tasm
Drive C is mounted as local directory c:\tasm\

Z:\>c:

C:\>debug
-o 43 B6
-o 42 77
-o 42 00
-o 61 3
-o 61 0
-q

C:\>debug
-o 43 b6
-o 42 a6
-o 42 04
-o 61 3
-o 61 0
-S_
```

Figure 4 (F out = 1 KHZ)

3.2.2. Part A.6.2 : Frequency = 5 KHZ

counter 2 = $F_{in} / F_{out} = 1.19 \text{ MHz} / 5 \text{ KHz} = EEh$.

```
C:\>debug
-o 43 b6
-o 42 ee
-o 42 00
-o 61 3
-o 61 0
-S
```

Figure 5 (F out = 5 KHz)

3.2.3. Part A.6.2 : Frequency = 15 KHZ

counter 2 = F in / F out = 1.19 MHz / 15 KHz = 4Fh

```
C:\>debug
-o 43 b6
-o 42 4f
-o 42 00
-o 61 3
-o 61 0
-S
```

Figure 6 (F out = 15 KHz)

3.3. Part B:- Using TASM to produce beep sounds

The code of part B in appendix (A.1) follows the same procedure as part A, B6 was put in the CR for PPI and the counter value which was 200D was put in counter 2, then the CR for PIT was configured to enable gate 2 and the speaker.

3.3.1. Task B.3:

Change the code to produce a beep sound of 3 KHz for 5 seconds (approximately).

Counter value = $1.19\text{MHz} / 3\text{KHz} \approx 397$.

So 397D will be put in counter 2, and 500 in T as code in appendix (A.2) shows.

After tasm and tlink for the code, a beep sound was generated for 5 seconds.

```
C:\>tasm ex6b3.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International

Assembling file:  ex6b3.asm
Error messages:  None
Warning messages: None
Passes:          1
Remaining memory: 491k

C:\>tlink ex6b3.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International

C:\>ex6b3.exe

C:\>S_
```

Figure 7 (Task B.3 Tasm & Tlink)

3.3.2. Task B.4:

Change the code to produce a beep sound of 12 KHz for 2 seconds (approximately).

Counter value = $1.19\text{MHz} / 12\text{KHz} \approx 99.17 \approx 99$.

So 99D will be put in counter 2, and 200 in T as code in appendix (A.3) shows.

3.4. Part C:- Generate Music on your PC

Lyrics	Notes	Freq. (Hz)	Duration
hap	C4	262	½
py	C4	262	½
birth	D4	294	1
day	C4	262	1
to	F4	349	1
you	E4	330	2
hap	C4	262	½
py	C4	262	½
birth	D4	294	1
day	C4	262	1
to	G4	392	1
you	F4	349	2
hap	C4	262	½
py	C4	262	½
birth	C5	523	1
day	A4	440	1
dear	F4	349	1
so	E4	330	1
so	D4	294	3
hap	B4b	466	½
py	B4b	466	½
birth	A4	440	1
day	F4	349	1
to	G4	392	1
you	F4	349	2

Figure 8 (Some Piano Notes And Frequencies)

We were asked to complete Part C code which is in the appendix (B.1) to play happy birthday song, The code use MACRO method to assign the required value to port b and counter and data to the speaker, the following Figure 9 show Tasm & Tlink for the code :-

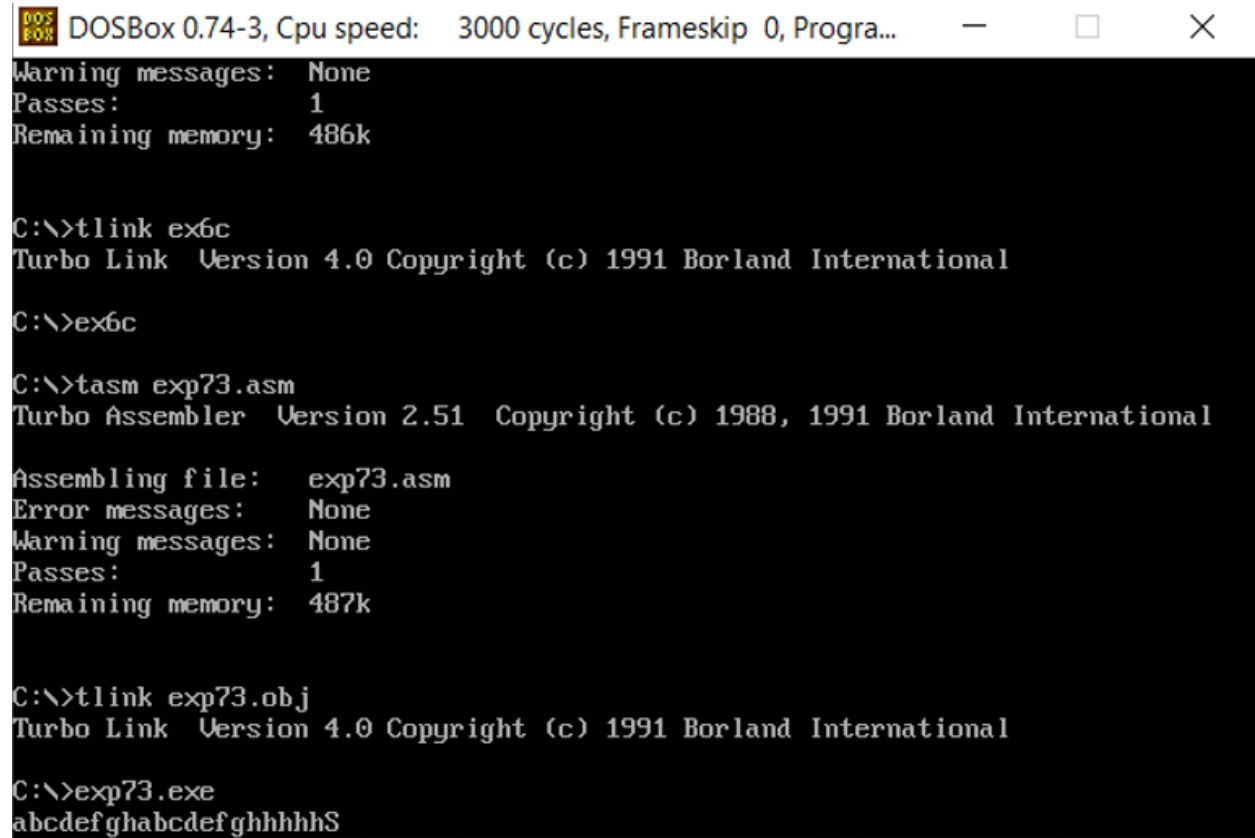
```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
C:\>tlink ex6b3.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
C:\>ex6b3.exe
C:\>tasm ex6c.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: ex6c.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 486k
C:\>tlink ex6c
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
C:\>tlink ex6c.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
C:\>ex6c.exe
C:\>S_
```

Figure 9 (Part C Tasm & Tlink)

3.5. Part D:- Using Keyboard as Piano keys

Using the keyboard keys, we were asked to play happy birthday song, the code in the appendix (C.1)

To play it the keys in the figure below should be pressed in the same order



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Warning messages: None
Passes: 1
Remaining memory: 486k

C:\>tlink ex6c
Turbo Link Version 4.0 Copyright (c) 1991 Borland International

C:\>ex6c

C:\>tasm exp73.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International

Assembling file: exp73.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 487k

C:\>tlink exp73.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International

C:\>exp73.exe
abcdefghabcdefghhhhhS
```

Figure 10 (Part D Tasm & Tlink)

4. Conclusion

In this experiment, we learned about PPI, PIT, how they work, their addresses and how to configure them and their control registers to get the obtained results.

5. References

[1] Lab Manual [Accessed 10:30 Aug 2020].

6. Appendix

6.1. A.1

```
.MODEL SMALL
.STACK 1000H
.DATA
COUNT EQU 200D
T EQU 500
.CODE
START:
MOV AL,0B6H
OUT 43H,AL
MOV AX,COUNT
OUT 42H,AL
MOV AL,AH
OUT 42H,AL
MOV AL, 00000011B
OUT 61H,AL
MOV CX,T
DELAY1:
PUSH CX
MOV CX,20000
DELAY2:
LOOP DELAY2
POP CX
LOOP DELAY1
```

```
MOV AL,00000000B;DISABLE GATE
OUT 61H,AL
MOV AX,4C00H
INT 21H
END START
```

6.2. A.2

```
.MODEL SMALL
.STACK 1000H
.DATA
COUNT EQU 397D
T EQU 500
.CODE
START:
MOV AL,0B6H
OUT 43H,AL
MOV AX,COUNT
OUT 42H,AL
MOV AL,AH
OUT 42H,AL
MOV AL, 00000011B
OUT 61H,AL
MOV CX,T
DELAY1:
PUSH CX
MOV CX,20000
DELAY2:
```

```
LOOP DELAY2
POP CX
LOOP DELAY1
MOV AL,00000000B;DISABLE GATE
OUT 61H,AL
MOV AX,4C00H
INT 21H
END START
```

6.3. A.3

```
.MODEL SMALL
.STACK 1000H
.DATA
COUNT EQU 99D
T EQU 200
.CODE
START:
MOV AL,0B6H
OUT 43H,AL
MOV AX,COUNT
OUT 42H,AL
MOV AL,AH
OUT 42H,AL
MOV AL, 00000011B
OUT 61H,AL
MOV CX,T
```

```
DELAY1:  
PUSH CX  
MOV CX,20000  
DELAY2:  
LOOP DELAY2  
POP CX  
LOOP DELAY1  
MOV AL,00000000B;DISABLE GATE  
OUT 61H,AL  
MOV AX,4C00H  
INT 21H  
END START
```

6.4. B.1

```
MODEL SMALL
.STACK 1000H
.DATA
T EQU 50
.CODE
TONE MACRO DIV,DUR
MOV AL,0B6H
OUT 43H,AL
MOV AX,DIV
OUT 42H,AL
MOV AL,AH
OUT 42H,AL
MOV AL,00000011B
OUT 61H,AL
MOV CX,DUR
CALL DELAY1
MOV AL,00000000B
OUT 61H,AL
CALL DELAY2
ENDM
.STARTUP
TONE 4553,T ; HAP (C4)
TONE 4553,T ; PY (C4)
TONE 4057,2*T ; BIRTH (D4)
TONE 4553,2*T;C4
```


TONE 6409,2*T;F4

TONE 3606,4*T;E4

TONE 4553,2*T;C4

TONE 4553,2*T;C4

TONE 4057,2*T;D4

TONE 4553,2*T;C4

TONE 3035,2*T;G4

TONE 3409,4*T;F4

TONE 4553,T;C4

TONE 4553,T;C4

TONE 2275,2*T;C5

TONE 2704,2*T;A4

TONE 3409,2*T;F4

TONE 3608,2*T;E4

TONE 4057,6*T;D4

TONE 2553,T;B4B

TONE 2553,T;B4B

TONE 2704,2*T;A4

TONE 3409,2*T;F4

TONE 3035,2*T;G4

TONE 3409,4*T;F4

MOV AH,4CH

INT 21H

DELAY1 PROC NEAR

D1:

PUSH CX

MOV CX,38000

```
D2:  
LOOP D2  
POP CX  
LOOP D1  
RET  
DELAY1 ENDP  
DELAY2 PROC NEAR  
MOV CX,65000  
D3:  
LOOP D3  
RET  
DELAY2 ENDP  
END
```

6.5. C.1

```
.model small
.stack 1000h
.data
t equ 50
.code
tone macro div,dur
mov al,0b6h
out 43h,al
mov ax,div
out 42h,al
mov al,ah
out 42h,al
mov al,00000011b
out 61h,al
mov cx,dur
call delay1
mov al,00000000b
out 61h,al
call delay2
endm
.startup
lab:
mov ah,1h
int 21h
cmp al,'a'
```

```
jz A
cmp al,'b'
jz B
cmp al,'c'
jz Ce
cmp al,'d'
jz n5
jmp s5
n5:
jmp D
s5:
cmp al,'e'
jz n1
jmp s1
n1:
jmp E
s1:
cmp al,'f'
jz n2
jmp s2
n2:
jmp F
s2:
cmp al,'g'
jz n3
jmp s3
n3:
```

jmp G

s3:

cmp al,'h'

jz n4

jmp s4

n4:

jmp H

s4:

A:

tone 2704,t;a4

jmp lab

B:

tone 2553,t;b4b

jmp lab

Ce:

tone 4553,t;c4

jmp lab

D:

tone 4057,t;d4

jmp lab

E:

tone 3608,t;e4

jmp lab

F:

tone 3409,t;f4

```
jmp lab
G:
tone 3035,t;g4
jmp lab
H:
tone 2275,t;c5
jmp lab
; Continue your code here.....
mov ah,4ch
int 21h
delay1 proc near
d1:
push cx
mov cx,38000
d2:
loop d2
pop cx
loop d1
ret
delay1 endp
delay2 proc near
mov cx,65000
d3:
loop d3
ret
delay2 endp
end
```