Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №3

«Программирование системного таймера»

Вариант 11

Проверил: Выполнил:  
к. т. н., доцент студент гр. 150501

Одинец Д. Н. Климович А. Н.

Минск, 2023

**1 ПОСТАНОВКА ЗАДАЧИ**

**Первая часть**: Запрограммировать второй канал таймера таким образом, чтобы динамик компьютера издавал звуки.

**Вторая часть**:

* Для всех каналов таймера считать слово состояния и вывести его на экран в двоичной форме.
* Для всех каналов таймера рассчитать коэффициент деления (значение счетчика CE) и вывести его на экран в шестнадцатеричной форме.

2 АЛГОРИТМ

Для того чтобы динамик компьютера издавал звуки, необходимо выполнить следующие действия:

1. Вывести в порт управляющего регистра управляющее слово, соответствующее каналу 2, режиму 3.
2. В порт 42h вывести значение делителя частоты.
3. Установить в 1 два младших бита порта 61h для включения звука.
4. Установить в 0 два младших бита порта 61h для выключения звука.

Для чтения слова состояния каналов необходимо:

1. Вывести в порт управляющего регистра управляющее слово, соответствующее команде RBC (*Чтение состояния канала)* и номеру канала.
2. Вывести из порта нужного канала слово состояния.

Для чтения текущего содержимого счетчика СЕ необходимо:

1. Вывести в порт управляющего регистра управляющее слово, соответствующее команде CLC (*Команда запоминания содержимого регистра СЕ*).
2. Вывести в порт управляющего регистра код команды запроса на чтение/запись в регистры канала (поле RW должно содержать 11).
3. Двумя последовательными командами ввода из порта нужного канала вывести младший и старший байты текущего состояния счетчика СЕ.

3 КОД ПРОГРАММЫ

Далее приведен листинг программы, которая выполняет поставленные задачи.

;LABA 3. FULL EDITION.

;VARIANT 11(1). "Fur elise" - Ludwig van Beethoven

;@AUTHOR - Klimovich Alexei, 150501

.model small

.stack 100h

.data

;divOfFreq = CR = 1193183/frequancy

frequancy dw 659, 622, 659, 622, 659, 493, 622, 523, 440

divOfFreq dw 712h,77Eh,712h,77Eh,712h,974h,77Eh,8E9h,0A97h

div\_factor dw ?

value\_of\_ce db ?

amountOfFreq equ 9

symbol db ?

counter db 9

\n db 0dh, 0ah, '$'

;MENU:

invite\_msg db "CHOOSE AN OPTION:", 0dh, 0ah, '$'

option\_1 db "1. Listen to music.", 0dh, 0ah, '$'

option\_2 db "2. Read state word and print on the screen in binary form.", 0dh, 0ah, '$'

option\_3 db "3. Calculate division coefficient.", 0dh, 0ah,'$'

option\_4 db "4. Calculate CE and print it in hex form", 0ah, 0dh, '$'

option\_5 db "5. Exit.", 0dh, 0ah, '$'

;INFORMATION STRINGS:

chanal\_0 db "40h - $"

chanal\_1 db "41h - $"

chanal\_2 db "42h - $"

music\_mode db ' ',0Eh,' ',0Eh,' ',0Eh,'$'

error\_msg db "ERROR!", 0dh, 0ah, '$'

continue\_msg db 0ah, 0dh, "Press any key to continue...", 0ah, 0dh, '$'

.code

;=======================================

;MACROS:

;---------------------------------------

PUSH\_ALL MACRO

push ax

push bx

push dx

push cx

ENDM

POP\_ALL MACRO

pop cx

pop dx

pop bx

pop ax

ENDM

PUTS\_STRING MACRO str

PUSH\_ALL

mov dx, offset str

mov ah, 09h

int 21h

POP\_ALL

ENDM

PUTCHAR MACRO symbol

PUSH\_ALL

mov ah, 02h

mov dl, symbol

int 21h

POP\_ALL

ENDM

GETCHAR MACRO symbol

PUSH\_ALL

mov ah, 08h ;;or you can use 01h too

int 21h

mov symbol, al

POP\_ALL

ENDM

SET\_FREQUENCY MACRO indexOfNota

push ax

mov al, BYTE ptr divOfFreq[indexOfNota]

out 42h, al

mov al, BYTE ptr divOfFreq[indexOfNota+1]

out 42h, al

pop ax

ENDM

GET\_DIV\_FACTOR MACRO freq

PUSH\_ALL

mov ax, 04DFh ;;1234DF = 1193183 Gz

mov dx, 0123h

mov bx, freq

div bx

mov div\_factor, ax

POP\_ALL

ENDM

;---------------------------------------

;FUNCTIONS:

;=======================================

PRINT\_DIV\_FACTOR PROC

PUSH\_ALL

mov ax, div\_factor ;our number

mov bx,16 ;base of counting system

xor cx,cx ;amount of digits in hex-number

L1:

xor dx,dx

div bx ;div number on 16

push dx ;save ostatok in stack

inc cx ;increment amoount of digits

test ax,ax ;does stay zero after division?

jnz L1 ;if not, then continue

L2:

pop ax ;take digit from stack

cmp al, 9 ;if digit > 9 (without sign)

ja isLetter

jmp isDigit

isLetter:

add al, 55 ;to letter

jmp print\_digit

isDigit:

add al,'0' ;transfer digit to ASCII symbol

print\_digit:

PUTCHAR al ;print digit

loop L2 ;all digits

POP\_ALL

ret

ENDP

;=======================================

PRINT\_REGISTER\_CE PROC

PUSH\_ALL

xor ax, ax

mov al,value\_of\_ce ;our number

mov bx,16 ;base of counting system

mov cx, 2 ;digits number is 2

LoopCE\_1:

xor dx,dx

div bx ;div number on 16

push dx ;save ostatok in stack

loop LoopCE\_1

mov cx, 2

jmp L2

ret

ENDP

;=======================================

PRINT\_BINARY\_NUMBER PROC

PUSH\_ALL

mov ax, bx ;our number

mov bx,2 ;base of counting system

xor cx,cx ;amount of digits in hex-number

mov cx, 8

Loop1:

xor dx,dx

div bx ;div number on 2

push dx ;save ostatok in stack

loop Loop1

mov cx, 8

Loop2:

pop ax ;take digit from stack

add al,'0' ;transfer digit to ASCII symbol

PUTCHAR al ;print digit

loop Loop2 ;all digits

POP\_ALL

ret

ENDP

;=======================================

PAUSE PROC

PUSH\_ALL ;200000 mks = 30D40 mks

mov dx, 0D40h

mov cx, 3h

mov ah, 86h

int 15h

POP\_ALL

ret

ENDP

;=======================================

CLEAR\_SCREEN proc ;This proc clears the user's screen

PUSH\_ALL

mov ax, 0600h

mov bh, 07h

xor cx, cx

mov dx, 184Fh

int 10h

;set cursor at (0;0)

xor dx, dx

xor bx, bx

xor ax, ax

mov ah, 2 ;Interrupt service 10h to change cursor position

int 10h

POP\_ALL

ret

endp

;=======================================

SPEAKER\_ON PROC

push ax

in al, 61h ;read state of port

or al, 00000011b ;set 1 in the first and second bits

out 61h, al ;turn on system speaker

pop ax

ret

ENDP

;=======================================

SPEAKER\_OFF PROC

push ax

in al, 61h ;read state of port

and al, 11111100b ;set 0 in the first and second bits

out 61h, al ;turn off system speaker

pop ax

ret

ENDP

;=======================================

PLAY\_SOUND PROC

cli ;block interruptions

call SPEAKER\_ON

call PAUSE

call SPEAKER\_OFF

sti

ret

ENDP

;=======================================

EXIT proc

mov ax,4C00h

int 21h

ret

endp

;---------------------------------------

; MAIN FUNCTION

;=======================================

start:

mov ax,@data

mov ds,ax

choice\_loop:

PUTS\_STRING invite\_msg

PUTS\_STRING option\_1 ;Listen to music

PUTS\_STRING option\_2 ;Read state word and print on the screen in binary form

PUTS\_STRING option\_3 ;Calculate division coefficient

PUTS\_STRING option\_4 ;Calculate CE and print it in hex form

PUTS\_STRING option\_5 ;Exit.

GETCHAR symbol

mov al, symbol

cmp al, '1'

je music\_option

cmp al, '2'

je stateWord\_option

cmp al, '3'

je divCoeff\_option

cmp al, '4'

je valueOfCE\_option

cmp al, '5'

je exit\_option

PUTS\_STRING error\_msg

loopne choice\_loop

;------------------------

music\_option:

call CLEAR\_SCREEN

PUSH\_ALL ;initialization:

mov al, 10110110b ;10 - third chanel

out 43h, al ;11 - read/write little byte, then high byte of CR(control register?)

xor bx, bx ;011 - mode of generetion meadra, positive and negative output signal take  
 ;part on the coordinate line

xor cx, cx ;0 - data presetation format(binary or hex)

mov cl, amountOfFreq

Next\_nota:

PUTS\_STRING music\_mode

SET\_FREQUENCY bx

call PLAY\_SOUND

PUTS\_STRING \n

inc bx

inc bx

loop Next\_nota

call PAUSE

POP\_ALL

call CLEAR\_SCREEN

jmp choice\_loop

;------------------------

stateWord\_option:

call CLEAR\_SCREEN

PUSH\_ALL

PUTS\_STRING chanal\_0

mov al, 11000010b ;TABLE 10.5 (Nesvizhskiy V.)

out 43h, al ;11 -command of reading registers of the chanels

in al, 40h ;0 - fix value of counter

mov bl, al ;0 - to get state of chanel

call PRINT\_BINARY\_NUMBER;001 - number of chanel (now chanel number is 0)

PUTS\_STRING \n ;0 - not used.

PUTS\_STRING chanal\_1

mov al, 11000100b ;TABLE 10.5 (Nesvizhskiy V.)

out 43h, al ;Analogically to the first chanel,

in al, 41h ;but number of chanel is

mov bl, al ;010 (now chanel number is 1)

call PRINT\_BINARY\_NUMBER

PUTS\_STRING \n

PUTS\_STRING chanal\_2

mov al, 11001000b ;TABLE 10.5 (Nesvizhskiy V.)

out 43h, al ;Analogically. Number of chanel is

in al, 42h ;100 (now chanel number is 2)

mov bl, al

call PRINT\_BINARY\_NUMBER

jmp end\_of\_operation

;------------------------

divCoeff\_option:

call CLEAR\_SCREEN

PUSH\_ALL

xor bx, bx

mov counter, 9

mov cl, amountOfFreq

print\_coeff:

mov div\_factor, 00000h

GET\_DIV\_FACTOR frequancy[bx]

call PRINT\_DIV\_FACTOR

PUTCHAR 20h ;space

inc bx

inc bx

dec counter

mov al, counter

cmp al, 0

je end\_of\_operation

jmp print\_coeff

;------------------------

valueOfCE\_option:

call CLEAR\_SCREEN

PUSH\_ALL

xor ax, ax

PUTS\_STRING chanal\_0

mov al, 00000000b ;TABLE 10.2-10.4 and 10.6

out 43h, al ;00 - chanel number (now is 0), other must be zero

in al, 40h ;little part of CE

push ax

in al, 40h ;high part of CE

mov value\_of\_ce, al

call PRINT\_REGISTER\_CE

pop ax

mov value\_of\_ce, al

call PRINT\_REGISTER\_CE

PUTS\_STRING \n

PUTS\_STRING chanal\_1

mov al, 01000000b ;TABLE 10.2-10.4 and 10.6

out 43h, al ;01 - chanel number (now is 1), other must be zero

in al, 41h ;little part of CE

push ax

in al, 41h ;high part of CE

mov value\_of\_ce, al

call PRINT\_REGISTER\_CE

pop ax

mov value\_of\_ce, al

call PRINT\_REGISTER\_CE

PUTS\_STRING \n

PUTS\_STRING chanal\_2

mov al, 10000000b ;TABLE 10.2-10.4 and 10.6

out 43h, al ;10 - chanel number (now is 2), other must be zero

in al, 42h ;little part of CE

push ax

in al, 42h ;high part of CE

mov value\_of\_ce, al

call PRINT\_REGISTER\_CE

pop ax

mov value\_of\_ce, al

call PRINT\_REGISTER\_CE

end\_of\_operation:

PUTS\_STRING continue\_msg

GETCHAR symbol

POP\_ALL

call CLEAR\_SCREEN

jmp choice\_loop

;------------------------

exit\_option:

call CLEAR\_SCREEN

call EXIT

end start

4 ТЕСТИРОВАНИЕ

На рисунках 4.1 – 4.3 приведены скриншоты результата работы программы.

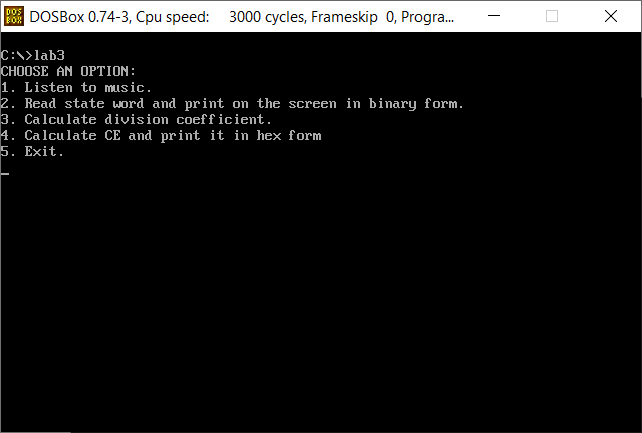


Рисунок 4.1 – Меню программы при запуске

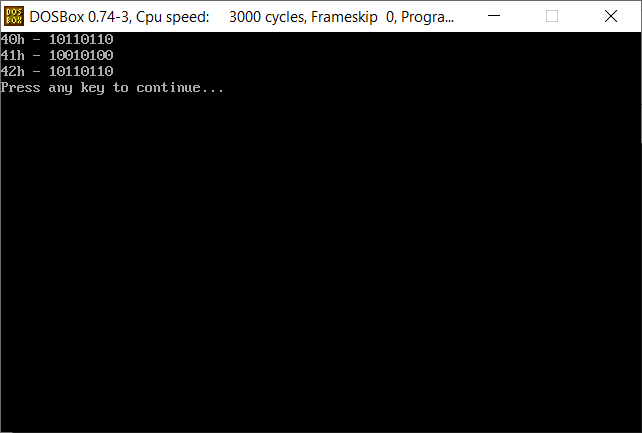


Рисунок 4.2 – Результат работы программы при выполнении опции “2”

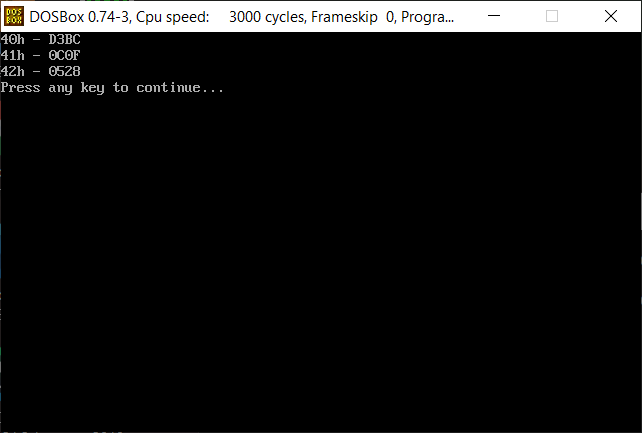


Рисунок 4.3 – Результат работы программы при выполнении опции “4”

5 ЗАКЛЮЧЕНИЕ

В ходе лабораторной был запрограммирован третий таймера с целью воспроизведения системным динамиков звуков. Для всех каналов таймера было считано слово состояния и выведено на экран в двоичной форме.   
Также, для всех каналов таймера был рассчитан коэффициент деления   
(значение счетчика CE) и выведен на экран в шестнадцатеричной форме.

Программа компилировалась в Turbo C++, эмуляция DOS была   
проведена с помощью DOSBox 0.74-3 на хосте 64-ех разрядной Windows 10.