Министерство образования Республики Беларусь

Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Кафедра ЭВМ

Отчёт по лабораторной работе №3

“Программирование системного таймера”

Проверил: Выполнил:

к.т.н., доцент студент гр.150501

Одинец Дмитрий Николаевич Черноок А.Ю.

Минск 2023

**Задача**

Целью данной работы является написание программы, работающей с системным таймером и динамиком.

**Алгоритм**

Программа состоит из 3-х частей:

* Воспроизведение мелодии с помощью динамика
* Вывод слов состояния в двоичной форме для каждого из каналов таймера;
* Вывод значений счетчиков CE для каждого из каналов таймера.

Воспроизведение мелодии происходит с использованием порта динамика 0x61. Работа с самим таймером происходит с использованием портов: 0x40 – для канала системных часов, 0x41 – для канала – контроллера прямого доступа к памяти, 0x42 – для канала – связи с системным динамиком, 0х43 – связи с управляющим регистром.

**Листинг программы**

#include <dos.h>

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <windows.h>

#define c 523

#define C 554

#define d 587

#define D 622

#define e 660

#define f 698

#define F 740

#define g 784

#define G 830

#define a 880

#define A 932

#define b 988

#define ONE 2000

unsigned int notes[] = { d / 2 , d / 2 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 , c / 2 , c / 2 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 };

unsigned int secs[] = { ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16 };

unsigned int notes1[] = { b / 4 , b / 4 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 , A / 4 , A / 4 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 };

unsigned int secs1[] = { ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16 };

//unsigned int notes2[] = { d / 2 , d / 2 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 , c / 2 , c / 2 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 };

//unsigned int secs2[] = { ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16 };

//unsigned int notes3[] = { b / 4 , b / 4 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 , A / 4 , A / 4 , d , 0 , 0 , a / 2 , 0 , 0 , 0 , G / 2 , 0 , g / 2 , 0 , f / 2 , d / 2 , f / 2 , g / 2 };

//unsigned int secs3[] = { ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16 };

unsigned int notes2[] = { d , d , d \* 2 , 0 , 0 , a , 0 , 0 , 0 , G , 0 , g , 0 , f , d , f , g , c , c , d \* 2 , 0 , 0 , a , 0 , 0 , 0 , G , 0 , g , 0 , f , d , f , g };

unsigned int secs2[] = { ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16 };

unsigned int notes3[] = { b / 2 , b / 2 , d \* 2 , 0 , 0 , a , 0 , 0 , 0 , G , 0 , g , 0 , f , d , f , g , A / 2 , A / 2 , d \* 2 , 0 , 0 , a , 0 , 0 , 0 , G , 0 , g , 0 , f , d , f , g };

unsigned int secs3[] = { ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 32, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 16, ONE / 8, ONE / 16, ONE / 16, ONE / 16 };

void speakerMod (char sOn) {

//DWORD dwResult = 0;

//inPort(0x61, &dwResult, 1);

int portValue = inp(0x61);

if (sOn) {

portValue |= 0x03;

//outPort(0x61, dwResult, 1);

}

else {

portValue &= 0xFC;

//outPort(0x61, dwResult, 1);

}

outp(0x61, portValue);

}

//void setFrequence(unsigned int uFrequence) {

// DWORD dwResult = 0;

// if (uFrequence < 0)

// return;

// int timerClock = 1193180;

// int iValue = 0;

//

// iValue = timerClock/uFrequence;

//

// outPort(0x43, 0xB6, 1);

// dwResult = iValue % 256;

// //dwResult = nDivider % 256;

//

// outPort(0x42, dwResult, 2);

// dwResult = iValue / 256;

// //dwResult = nDivider / 256;

//

// outPort(0x42, dwResult, 2);

//}

void setFrequence(unsigned int \_freq) {

//DWORD dwResult = 0;

//if (\_freq < 0)

// return;

long timerClock = 1193180;

int divValue = 0;

divValue = timerClock/\_freq;

//outPort(0x43, 0xB6, 1);

outp(0x43, 0xB6);

//dwResult = nDivider % 256;

int freqParam = divValue % 256;

outp(0x42, freqParam);

//dwResult = nDivider / 256;

freqParam = divValue / 256;

outp(0x42, freqParam);

return;

}

void music\_play() {

int i = 0;

for (; i < 34; i++) {

if (kbhit()) {

//printf("!!!!!!!!!!!!");

getch();

break;

}

if (notes[i] == 0)

delay(secs[i]);

else {

setFrequence(notes[i]);

speakerMod(1);

delay(secs[i]);

speakerMod(0);

}

delay(20);

//delsecs[i];

}

}

void long\_music\_play() {

int i = 0;

for (; i < 34; i++) {

if (kbhit()) {

getch();

break;

}

if (notes[i] == 0)

delay(secs[i]);

else {

setFrequence(notes[i]);

speakerMod(1);

delay(secs[i]);

speakerMod(0);

}

delay(20);

//delsecs[i];

}

i = 0;

for (; i < 34; i++) {

if (kbhit()) {

getch();

break;

}

if (notes1[i] == 0)

delay(secs1[i]);

else {

setFrequence(notes1[i]);

speakerMod(1);

delay(secs1[i]);

speakerMod(0);

}

delay(20);

//delsecs[i];

}

i = 0;

for (; i < 34; i++) {

if (kbhit()) {

getch();

break;

}

if (notes2[i] == 0)

delay(secs2[i]);

else {

setFrequence(notes2[i]);

speakerMod(1);

delay(secs2[i]);

speakerMod(0);

}

delay(20);

//delsecs[i];

}

i = 0;

for (; i < 34; i++) {

if (kbhit()) {

getch();

break;

}

if (notes3[i] == 0)

delay(secs3[i]);

else {

setFrequence(notes3[i]);

speakerMod(1);

delay(secs3[i]);

speakerMod(0);

}

delay(20);

}

}

void bin\_print(unsigned value) {

char bin\_value[] = "00000000\0";

char temp;

for (int i = 7; i >= 0; i--)

{

temp = value % 2;

value /= 2;

bin\_value[i] = temp + '0';

}

printf("%sb", bin\_value);

}

void get\_stateWord() {

printf("\nTimer State Words:\n");

//char state;

unsigned state;

printf("Channel 0: ");

outp(0x43, 0xE2); // command rbc for reading state word 0xE2 = 11100010B

state = inp(0x40);

printf("%4Xh ---- ", state);

bin\_print(state);

printf("\nChannel 1: ");

outp(0x43, 0xE4); // 0xE4 = 11100100B

state = inp(0x41);

printf("%4Xh ---- ", state);

bin\_print(state);

printf("\nChannel 2: ");

outp(0x43, 0xE8); // 0xE8 = 11101000B

state = inp(0x42);

printf("%4Xh ---- ", state);

bin\_print(state);

getch();

}

void get\_counterValue() {

printf("\nTimer Channels Counter CE Values:");

unsigned state;

printf("\nChannel 0: "); // output the CLC command code to the control register port - 00000000b

outp(0x43, 0x00); // output the command code of the read/write request command

// to the control register port - 00110000b

outp(0x43, 0x30); // get the counter CE current state lowest byte

state = inp(0x40); // get and add the counter CE current state highest byte

state = (inp(0x40) << 8) + state;

printf("%04Xh", state);

printf("\nChannel 1: "); // output the CLC command code to the control register port - 01000000b

outp(0x43, 0x40); // output the command code of the read/write request command

// to the control register port - 01110000b

outp(0x43, 0x70); // get the counter CE current state lowest byte

state = inp(0x41); // get and add the counter CE current state highest byte

state = (inp(0x41) << 8) + state;

printf("%04Xh", state);

printf("\nChannel 2: "); // output the CLC command code to the control register port - 10000000b

outp(0x43, 0x80); // output the command code of the read/write request command

// to the control register port - 10110000b

outp(0x43, 0xB0); // get the counter CE current state lowest byte

state = inp(0x42); // get and add the counter CE current state highest byte

state = (inp(0x42) << 8) + state;

printf("%04Xh", state);

getch();

}

int main(){

char ch;

while (1) {

clrscr();

printf("Choose option:\n'1' - Music\n'2' - Music Long Version\n'3' - Get & Print Timer State Words\n'4' - Get & Print Counter CE Value\n'0' - End\n");

ch = getch();

switch (ch)

{

case '1':

music\_play();

break;

case'2':

long\_music\_play();

break;

case'3':

get\_stateWord();

break;

case'4':

get\_counterValue();

break;

default:

break;

}

if (ch == '0')

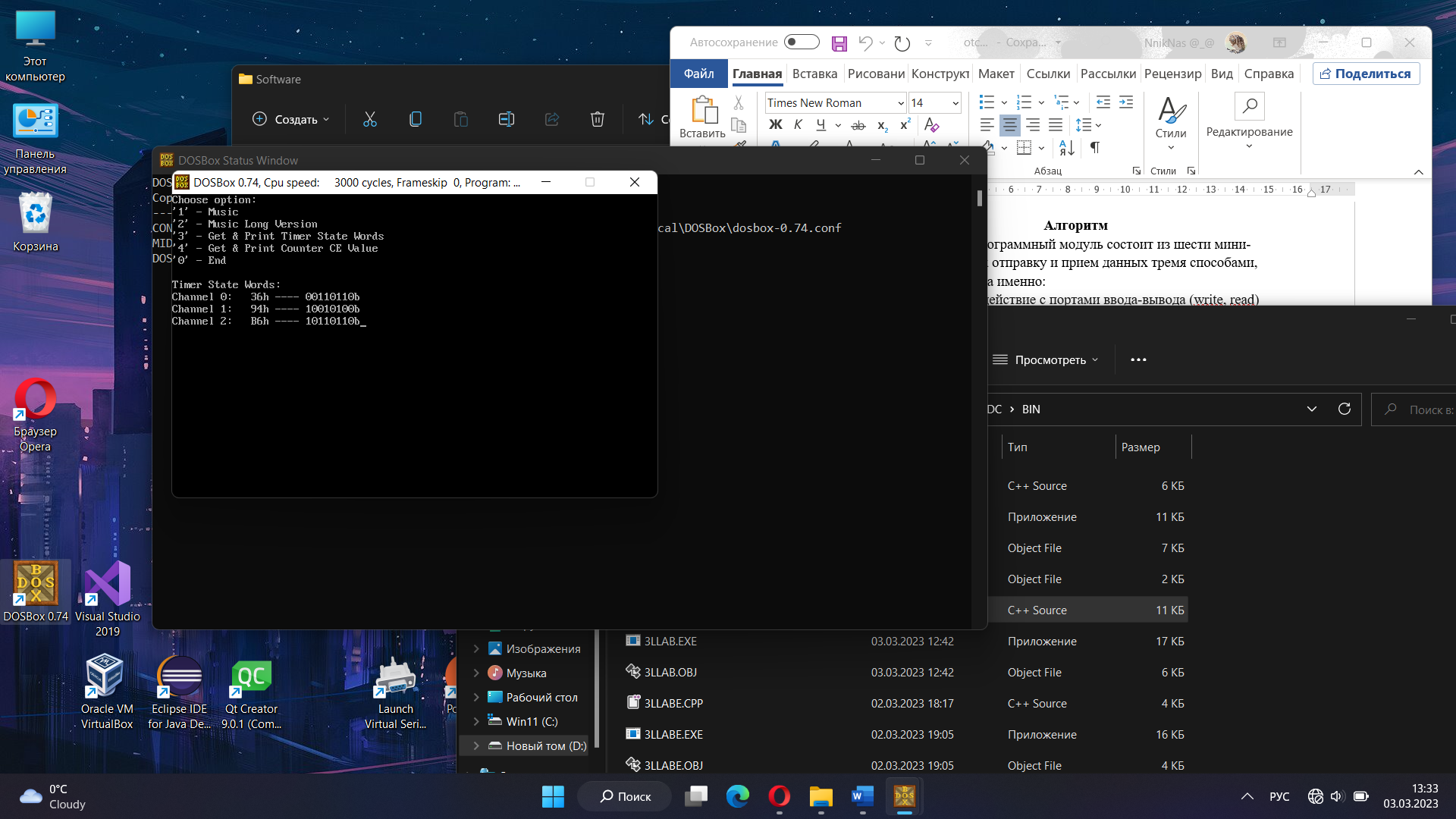
break;

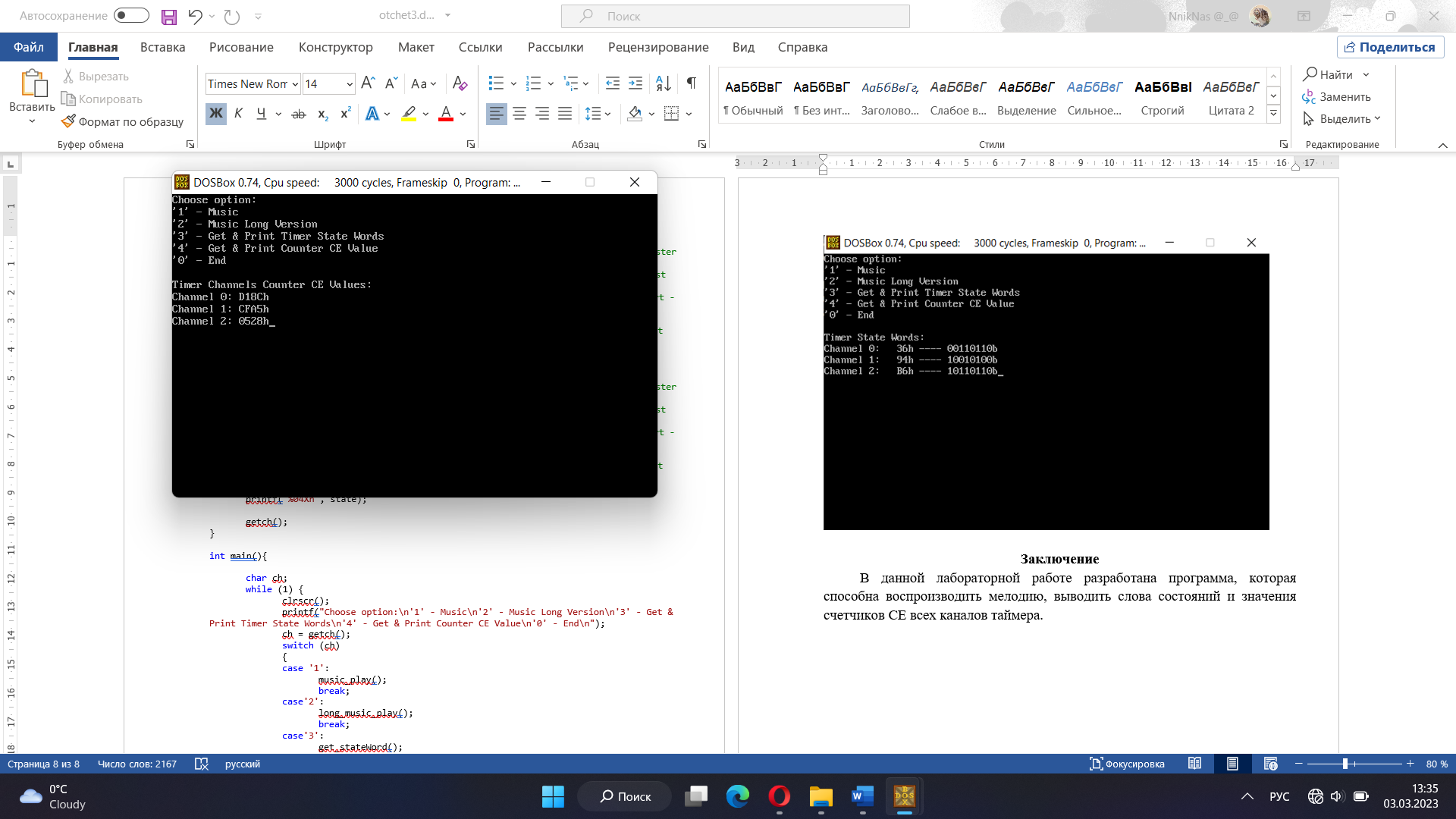
}

return 0;

}

**Тест**





**Заключение**

В данной лабораторной работе разработана программа, которая способна воспроизводить мелодию, выводить слова состояний и значения счетчиков СЕ всех каналов таймера.