Министерство образования Республики Беларусь
Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»
Кафедра ЭВМ

Лабораторная работа № 2
«Системный таймер персонального компьютера»

Выполнил:
студент группы 724402
Чернявский Я.А.
Проверил:
к.т.н., доцент Селезнёв И.Л.

Минск
2019

1. Цель работы

1) Получить навыки программирования и изучить возможности использования системного таймера.
2) С помощью системного таймера сгенерировать мелодию, состоящую из нот заданной частоты и длительности и вывести ее на динамик компьютера (мелодия подбирается самостоятельно, в группе не должно быть дублей, длительность - от 15 сек, частотный диапазон желателен 300 - 1300 Гц).
3) Вывести словосостояния для каждого канала в двоичном виде и шестнадцатеричном виде.

2. Порядок выполнения работы

1) Произвести настройку регистра управления системного таймера.
2) Записать значение в порт динамика.
3) Вывести звуковой сигнал.
4) После проигрывания всей мелодии вывести слово состояния для каждого канала в двоичном и шестнадцатеричном виде.

3. Выполнение работы

Системный таймер - устройство, подключенное к линии запроса IRQ0 и вырабатывающее прерывание int8h. Таймер реализуется на микросхеме Intel 8253 или 8254 (Отечественные аналоги: К1810ВИ53 и К1810ВИ54).

Таймер состоит из трёх независимых каналов: 0 - отсчитывает текущее время после включения компьютера, 1- для работы с контроллером прямого доступа к памяти, 2 - связан с системным динамиком.

Каждый канал содержит регистры:

- состояния канала RS (8 разрядов);
- управляющего слова RSW (8 разрядов);
- буферный регистр OL (16 разрядов);
- регистр счетчика CE (16 разрядов);
- регистр констант пересчета CR (16 разрядов)

Каналы таймера подключаются к внешним устройствам при помощи линий:

- GATE - управляющий вход;
- CLOCK - вход тактовой частоты;
- OUT - выход таймера.

Регистр счетчика CE работает в режиме вычитания. Его содержимое уменьшается по спаду сигнала CLOCK при условии, что на вход GATE установлен уровень логической 1.

В зависимости от режима работы таймера при достижении счетчиком CE нуля тем или иным образом изменяется выходной сигнал OUT.

Буферный регистр OL предназначен для запоминания текущего содержимого регистра счетчика CE без остановки процесса счета. После запоминания буферный регистр доступен программе для чтения. Регистр констант пересчета CR может загружаться в регистр счетчика, если это требуется в текущем режиме работы таймера.

Таймер позволяет воспроизводить звуки в фоновом режиме. Канал 2 микросхемы 8254 связан с громкоговорителем компьютера. Однако громкоговоритель не просто соединен с выходом OUT канала 2. Порт вывода 61h также используется для управления громкоговорителем. Младший бит порта 61h подключен ко входу GATE канала 2 таймера. Этот бит при установке в 1 разрешает работу канала, т.е. если этот бит установлен в 1, импульсы от канала 2 таймера смогут проходить на громкоговоритель.

Бит 0 фактически разрешает работу данного канала таймера, а бит 1 включает динамик.

Возможны шесть режимов работы таймера, которые делятся на три типа:

Режимы 0, 4 - однократное выполнение функций.

Режимы 1, 5 - работа с перезапуском.

Режимы 2, 3 - работа с автозагрузкой.

Режим однократного выполнения функций

В режиме однократного выполнения функций перед началом счета содержимое регистра констант пересчета CR переписывается в регистр счетчика CE по сигналу CLOCK, если сигнал GATE установлен в 1. В дальнейшем содержимое регистра CE уменьшается по мере прихода импульсов CLOCK. Процесс счета можно приостановить, если подать на вход GATE уровень логического 0. Если затем на вход GATE подать 1, счет будет продолжен дальше. Для повторения выполнения функции необходима новая загрузка регистра CR, то есть повторное программирование таймера.

Работа с перезапуском

При работе с перезапуском не требуется повторного программирования таймера для выполнения той же функции. По фронту сигнала GATE значение константы из регистра CR вновь переписывается в регистр CE, даже если текущая операция не была завершена.

Режим автозагрузки

В режиме автозагрузки регистр CR автоматически переписывается в регистр CE после завершения счета. Сигнал на выходе OUT появляется только при наличии на входе GATE уровня логической 1. Этот режим используется для

создания программируемых импульсных генераторов и генераторов прямоугольных импульсов (меандра).

Для подключения динамика используется порт 61h

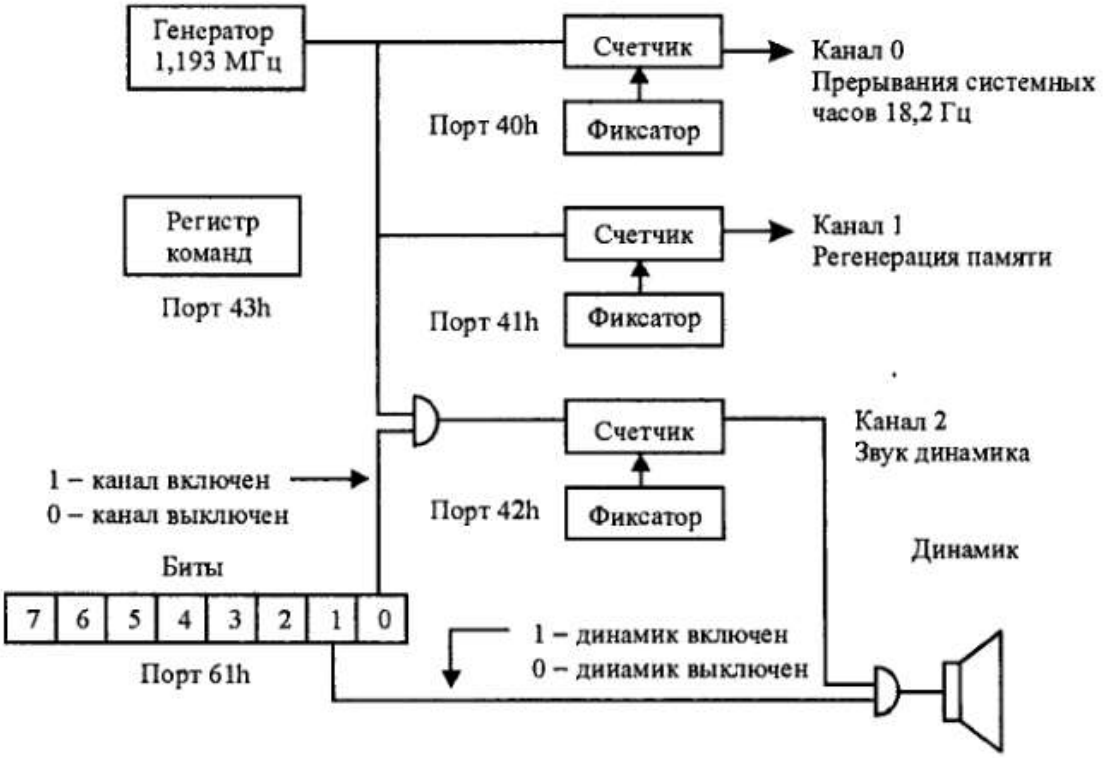

Рисунок 3.1 – Схема подключения таймера к динамику

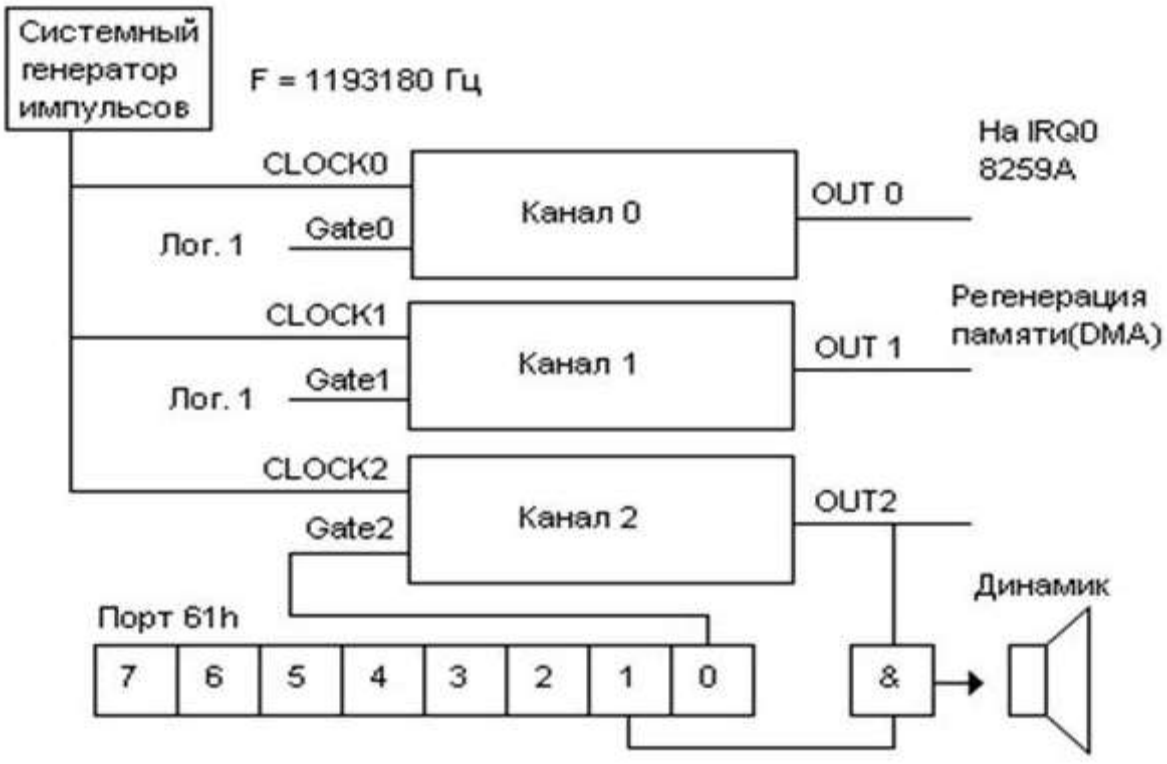

Рисунок 3.2 – Схема подключения таймера к динамику

Биты	7	6	5	4	3	2	1	0
Описание	1	1	Б	Статус	Канал 2	Канал 1	Канал 0	0

Приведем краткое описание таблицы.

- Бит 0 не используется и должен быть установлен в 0.
- Биты 1—3 позволяют установить номера каналов. Установка бита в 1 выбирает соответствующий номер канала.
- Бит 4 позволяет получить состояние для выбранного канала (каналов) при установке в 0.
- Бит 5 позволяет зафиксировать значение счетчика для выбранного канала (каналов) при установке в 0.
- Биты 6 и 7 определяют команду чтения и должны быть установлены в 1.

При выполнении команды блокировки счетчика (биты 4 и 5 установлены в 0) можно получить значение (состояние) выбранного счетчика без остановки самого таймера. Результат считывается из указанного канала (биты 6 и 7). При этом формат команды будет таким, как показано в табл. 10.6.

Рисунок 3.3 – Формат управляющего регистра в режиме считывания

Биты	7	6	5	4	3	2	1	0
Описание	OUT	Готов	Тип операции		Режим работы			Формат

Приведем краткое пояснение к таблице 10.7.

- Бит 0 определяет формат представления данных: 0 — двоичный (16-битное значение от `0000h` до `FFFFh`), 1 — двоично-десятичный (BCD от `0000` до `9999`).
- Биты 1—3 определяют режим работы таймера. Возможные значения перечислены в табл. 10.2.
- Биты 4—5 определяют тип операции
- Бит 6 определяет готовность счетчика для считывания данных (1 — готов, 0 — счетчик обнулен).
- Бит 7 определяет состояние выходного сигнала на канале в момент блокировки счетчика импульсов.

Рисунок 3.4 – Формат байта состояния канала

3.1. Листинг кода программы

```
//724402-2
//Chernyavsky YA

//LR-2 System timer

#include <dos.h>

#include <conio.h>

#include <stdio.h>

#include <process.h>

#include <stdlib.h>

#include <string.h>

struct note {

	int tone;

	int delay;

};

void PlayNote(struct note n);

void ResetScreen(void);

void GetStateWord(void);

void setTonality(int freq);

void onsound(void);

void offsound(void);

char* fillBin(char* in);


int main()

{

	struct note song_notes[50] = {


		{392, 500},//g4

		{262, 500},//c4

		{330, 250},//e4

		{349, 250},//f4

		{392, 500},//g4

		{262, 500},//c4

		{330, 250},//e4
```

```
{349, 250},//f4

{392, 500},//g4
{262, 500},//c4
{311, 250},//ds4
{349, 250},//f4
{294, 500},//d4
{196, 500},//g3
{233, 250},//as3
{262, 250},//c4

{294, 500},//d4
{196, 500},//g3
{233, 250},//as3
{262, 250},//c4
{294, 500},//d4
{196, 500},//g3
{233, 250},//as3
{262, 250},//c4

{294, 500},//d4
{196, 500},//g3
{233, 250},//as3
{262, 250},//c4
{294, 1000},//d4
{349, 1000},//f4
{233, 1000},//as3
{311, 250},//ds4

{294, 250},//d4
{349, 1000},//f4
{233, 1000},//as3
{311, 250},//ds4
```

```
        {294, 250},//d4
        {262, 500},//c4
        {208, 250},//gs3
        {233, 250},//as3

        {262, 500},//c4
        {175, 500},//f3
        {208, 250},//gs3
        {233, 250},//as3
        {262, 500},//c4
        {175, 500},//f3
        {208, 250},//gs3
        {233, 250},//as3

        {262, 500},//c4
        {175, 500},//f3


};

int i, length = sizeof(song_notes) / sizeof(song_notes[0]);
char wait[4] = "-\\|/";
ResetScreen();

while (1)     //Меню программы
{
        printf("1. Get status word\n2. Play sound\n3. Exit\n");
        switch (getch())     //Считывание символа введенного с клавиатуры
        {
        case '1':
                GetStateWord();      //Вывод регистра словосостояния и 61h порта
                getch();
                ResetScreen();       //обновление экрана
```

```
                        break;
                case '2':
                        GetStateWord();
                        for (i = 0; i < length; i++)//Проигрывание мелодии
                        {
                                printf("(%i|%-4i)%c\b", song_notes[i].tone,
song_notes[i].delay, wait[i % 4]); //Вывод частоты и длительности ноты
                                onsound();                              //Включение динамика
                                PlayNote(song_notes[i]);    //Проигрывание ноты из массива
                                offsound();                             //Выключение
динамика
                        }
                        GetStateWord();
                        printf("press key...\n");
                        getch();
                        ResetScreen();       //Обновление экрана
                        break;
                case '3': case 27:   //Считан сивол 3 или Esc
                        return 0;            //Выход из программы
                default:                     //Считан любой другой символ
                        printf("Wrong code!\n");
                        getch();
                        ResetScreen();
                        break;
                }
        }
}
void ResetScreen() { //Функция обновления экрана
        system("cls");
        printf("%60s", "724402-2. Chernyavsky Y.A. Lab02 - System timer\n\n");
}
void setTonality(int freq) {       //установка тональности и проигрывание звука
        short value = 1193180 / freq;            //Частота таймера равна 1193180 Гц
        outp(0x42, (char)value);
```

```
	outp(0x42, (char)(value >> 8));
}
void onsound() {     //Включение динамика
	char port61;
	port61 = inp(0x61);
	port61 = port61 | 3;
	outp(0x61, port61);
}
void offsound() {    //Выключение динамика
	char port61;
	port61 = port61 & 0xFFFC;
	outp(0x61, port61);
}
void PlayNote(struct note n) {     //Функция проигрывания ноты
	setTonality(n.tone);
	delay(n.delay);
}

void GetStateWord() {       //Функция вывода регистра словосостояния и регистра порта 61h
	unsigned i;
	unsigned char temp;
	char* fill = (char*)calloc(9, sizeof(char));
	char* fill61 = (char*)calloc(9, sizeof(char));

	//считывание словосостояния канала 0
	outp(0x43, 0xe2);
	temp = inp(0x40);
	itoa(temp, fill, 2); //перевод словосостояния в двоичную систему
	fill = fillBin(fill);
	//вывод словосостояния канала 0
	printf("\nChannel 0 state word: %02.2X(h) | %8s(b)", inp(0x40), fill);

	//считывание словосостояния канала 1
```

```
        outp(0x43, 0xe4);
        temp = inp(0x41);
        itoa(temp, fill, 2);
        fill = fillBin(fill);
        //вывод словосостояния канала 1
        printf("\nChannel 1 state word: %02.2X(h) | %8s(b)", inp(0x41), fill);


        //считывание словосостояния канала 2
        outp(0x43, 0xe8);
        temp = inp(0x42);
        itoa(temp, fill, 2);
        temp = inp(0x61);    //считывание регистра порта h61
        itoa(temp, fill61, 2);
        fill = fillBin(fill);
        fill61 = fillBin(fill61);
        //вывод словосостояния канала 2
        printf("\nChannel 2 state word: %02.2X(h) | %8s(b)", inp(0x42), fill);
        //вывод регистра 61h порта
        printf("\nPort 61 register:     %02.2X(h) | %8s(b)\n", inp(0x61), fill61);
        free(fill);
}

char* fillBin(char* in) {   //Функция дополнения двоичных чисел нулями
        char* out = (char*)calloc(9, sizeof(char));
        int zamount = 8 - strlen(in), j;
        for (j = 0; j < zamount; j++) {
                out[j] = '0';
        }
        strcat(out, in);
        return out;}
```

3.2. Скриншоты выполнения программы

Ниже представлены скриншоты, демонстрирующие работу программы. На рисунке 3.2.1 –главное меню программы. На рисунке 3.2.2 – вывод функции отображения словосостояния каналов и порта 61h. На рисунке 3.2.3 – вывод функции проигрывания мелодии.

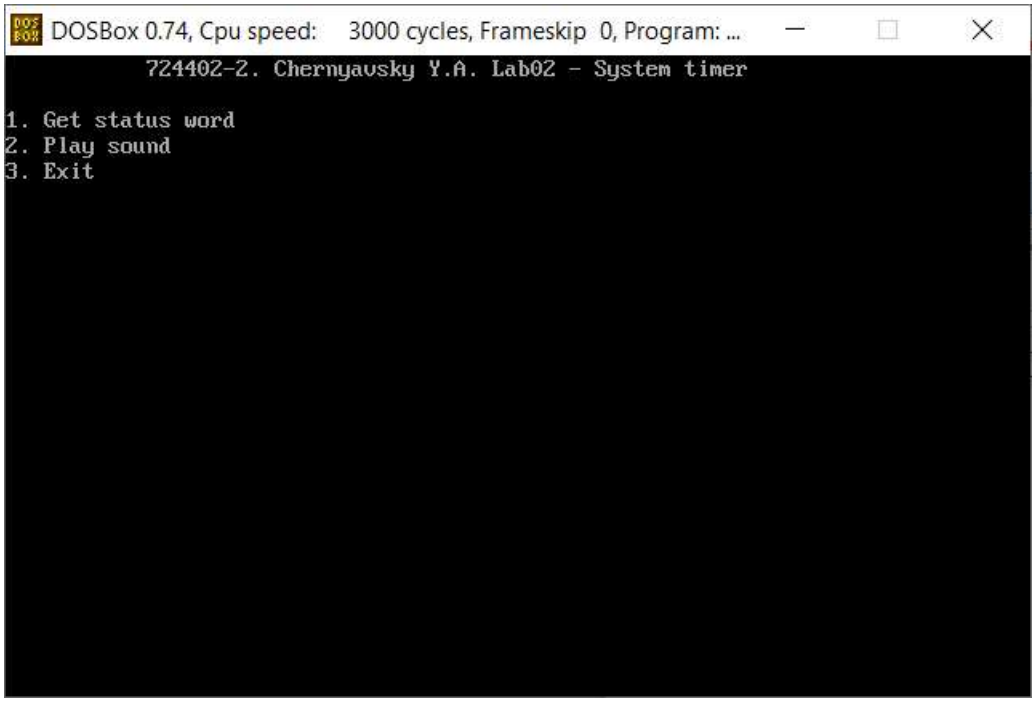

Рисунок 3.2.1 – Меню программы

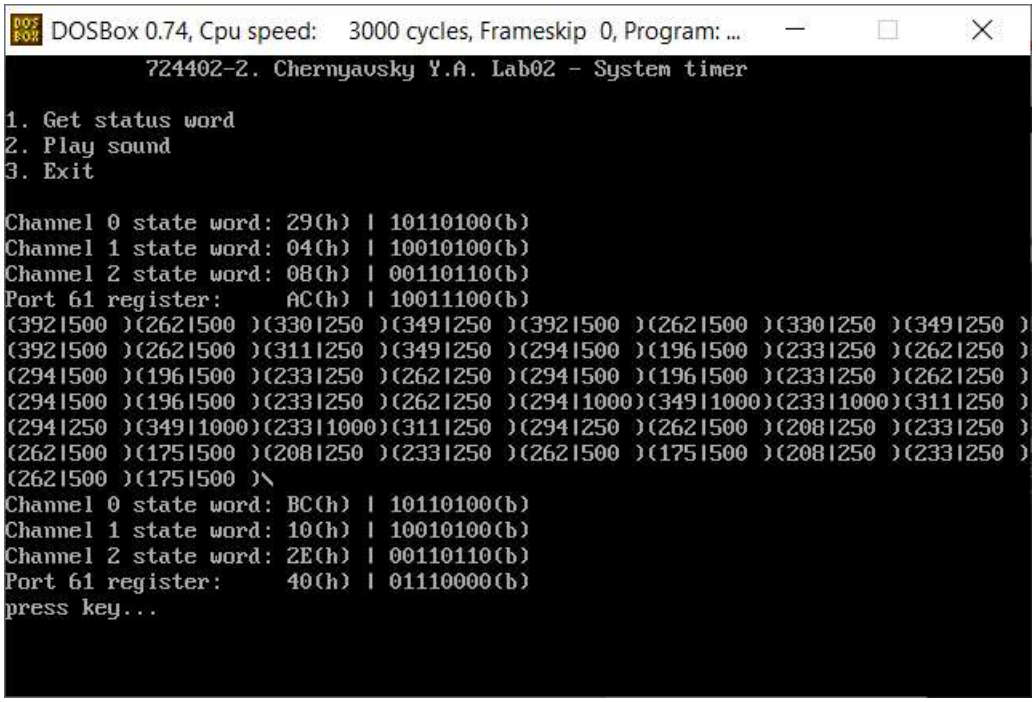

Рисунок 3.2.2 – Отображение состояния каналовРисунок 3.2.3 – Проигрывания мелодии

4. Вывод

В результате выполнения лабораторной работы были изучены способы управления каналами системного таймера, изучены режимы работы системного таймера, а также взаимодействие системного таймера с динамиком.