Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №3

«Программирование системного таймера»

Вариант 5

Выполнил: Проверил:

Студент группы 250502 Преподаватель

Бригадир А.С. Одинец Д.Н.

Минск, 2024

1. Постановка задачи

Запрограммировать второй канал таймера таким образом, чтобы динамик компьютера издавал звуки.

Для всех каналов таймера считать слово состояния и вывести его на экран в двоичной форме.

1. Алгоритм

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

* Вывести в порт управляющего регистра с адресом 43h управляющее слово 10110110, соответствующее каналу 2, режиму 3
* Установить значение счётчика канала 2 таймера: в порт 42h вывести значение, полученное при разделении 1193180 на требуемую частоту в герцах, причём вначале вывести младший, а затем старший байты.
* Установить в 1 два младших бита порта 61h для включения звука. Для этого вначале считывается байт из порта 61h в рабочую ячейку памяти, устанавливаются нужные биты, затем выводится новое значение байта в порт 61h.
* Установить в 0 два младших бита порта 61h для выключения звука.

Для чтения слова состояния каналов необходимо:

* Вывести в порт управляющего регистра с адресом 43h управляющее слово, соответствующее команде RBC (*Чтение состояния канала)* и номеру канала.
* Вывести из порта нужного канала слово состояния.

1. Листинг программы

#include <stdio.h>

#include <conio.h>

#include <dos.h>

#include <stdlib.h>

#include <math.h>

#include <malloc.h>

const int SOUND\_DURATION\_MILISECONDS = 50;

int multyplied=1;

void ReadStatusWords(void);

void EnableSound();

void DisableSound();

short HerzToShort(int);

void ChannelsKd();

void piano();

void main()

{

    char c=0;

    outp(0x43, 0xB6); // channel 2, operation 4, mode 3, format 0

    while(c!=27){

        printf("Press: \n 0 - play piano\n1 - read status words\n2 - read CE\nesc-exit\n");

        c = getch();

        switch(c){

            case '0':

                piano();

                clrscr();

                break;

            case '1':

                ReadStatusWords();

                break;

            case '2':

                ChannelsKd();

                break;

            case 27:

                break;

        }

    }

    return;

}

void EnableSound()

{

    outp(0x61, inp(0x61) | 3);

}

void DisableSound()

{

    char port61;

    port61 = port61 & 0xFFFC;

    outp(0x61, port61);

}

void SetTonality (int frequency)

{

    short value\_of\_CE = 1193180/frequency;

    outp(0x42, (char)value\_of\_CE);      // low byte

    outp(0x42, (char)(value\_of\_CE >> 8)); // high byte

}

void ReadStatusWords()

{

    char\* str = (char\*)calloc(9,1);

    unsigned char value;

    outp (0x43, 0xE2); // command to read first channel

    value = inp(0x40);

    itoa(value, str,2);

    printf ("first channel status word : %s\n", str);

    outp (0x43, 0xE4); // command to read second status word

    value = inp(0x41);

    itoa(value, str,2);

    printf ("second channel status word: %s\n", str);

    outp (0x43, 0xE8); // command to read second status word

    value = inp(0x42);

    itoa(value, str,2);

    printf ("third channel status word : %s\n", str);

    free (str);

}

void ChannelsKd()

{

    char \* str =  (char\*)calloc(9,1);

    char ce\_high =0;

    char ce\_low = 0;

    short result = 0;

    outp (0x43, 0xD2);

    ce\_low = inp(0x40);

    ce\_high = inp(0x40);

    result = ce\_high\*256+ce\_low;

    itoa(result, str,2);

    printf ("first channel CE value %s (%d)\n", str, result);

    outp (0x43, 0xD4);

    ce\_low = inp(0x41);

    ce\_high = inp(0x41);

    result = ce\_high\*256+ce\_low;

    itoa(result, str,2);

    printf ("first channel CE value %s (%d)\n", str, result);

    outp (0x43, 0xD8);

    ce\_low = inp(0x42);

    ce\_high = inp(0x42);

    result = ce\_high\*256+ce\_low;

    itoa(result, str,2);

    printf ("first channel CE value %s (%d)\n", str, result);

    free(str);

}

void piano(){

    char c=0;

    printf ("press keys to play:)\n");

    printf ("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

    printf ("|   |  |  |  |   |   |  |  |  |  |  |   |   |  |  |  |   |\n");

    printf ("|   |  |  |  |   |   |  |  |  |  |  |   |   |  |  |  |   |\n");

    printf ("|   | w|  | e|   |   | t|  | y|  | u|   |   | o|  | p|   |\n");

    printf ("|   |\_\_|  |\_\_|   |   |\_\_|  |\_\_|  |\_\_|   |   |\_\_|  |\_\_|   |\n");

    printf ("|    |      |    |    |     |      |    |    |      |    |\n");

    printf ("|  a |   s  |  d |  f |  g  |   h  |  j |  k |   l  |  ; |\n");

    printf ("|\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_|\n");

    while(c != 27)

    {

        c = getch();

        switch(c)

        {

            case 'a':

                SetTonality(131\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 's':

                SetTonality(294/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'd':

                SetTonality(330/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'f':

                SetTonality(349/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'g':

                SetTonality(392/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'h':

                SetTonality(440/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'j':

                SetTonality(494/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'k':

                SetTonality(131\*(multyplied\*2));

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'l':

                SetTonality(294/2\*(multyplied\*2));

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case ';':

                SetTonality(330/2\*(multyplied\*2));

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'w':

                SetTonality(277/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'e':

                SetTonality(311/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 't':

                SetTonality(370/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'y':

                SetTonality(415/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'u':

                SetTonality(466/2\*multyplied);

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'o':

                SetTonality(277/2\*(multyplied\*2));

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case 'p':

                SetTonality(311/2\*(multyplied\*2));

                EnableSound();

                delay(SOUND\_DURATION\_MILISECONDS);

                DisableSound();

                break;

            case '1':

                multyplied = 1;

                break;

            case '2':

                multyplied = 2;

                break;

            case '3':

                multyplied = 4;

                break;

            case '4':

                multyplied = 8;

                break;

            case 27:

                break;

        }

    }

    clrscr();

}

1. Тестирование программы

Во время работы программы происходит звучание системного динамика. Также для всех каналов таймера выводится на экран в двоичной форме слово состояния:

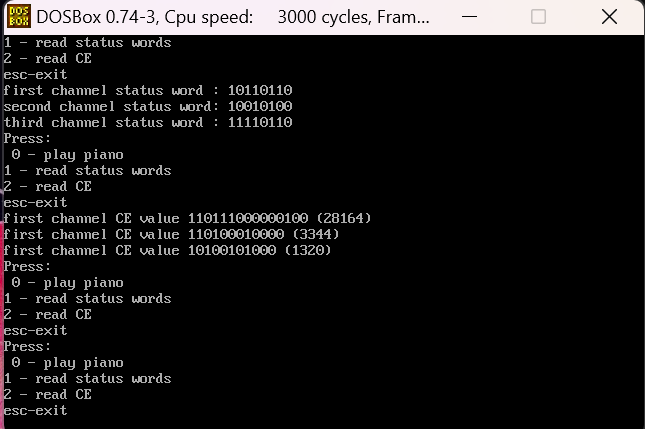


Рисунок 4.1 – Результат работы программы при выводе слов состояния каналов таймера.

1. Заключение

В ходе лабораторной работы удалось запрограммировать второй канал таймера таким образом, чтобы динамик компьютера издавал звук, а также для всех каналов таймера было считано слово состояния и выведено на экран в двоичной форме.

Программа компилировалась в Turbo C++ и запускалась в DOS, который эмулировался с помощью DosBox.