Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №3

«Программирование системного таймера»

Вариант 4

Выполнил: Проверил:

Студент группы 150504 Преподаватель

Желубовский С.В. Одинец Д.Н.

Минск, 2023

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

Далее приведен листинг программы, реализующей все поставленные задачи.

#include <stdio.h>

#include <conio.h>

#include <dos.h>

#define d2 73u

#define c3 130u

#define D3 156u

#define f3 175u

#define a3 220u

#define A3 233u

#define d4 294u

#define D4 311u

#define f4 349u

#define g4 392u

#define a4 440u

#define A4 466u

#define c5 523u

#define d5 587u

#define D5 622u

#define f5 698u

#define g5 784u

#define a5 880u

#define A5 932u

#define C6 1109u

#define e3 164u

#define e4 329u

#define duration 150u

#define indent 150u

#define NOTES\_AMOUNT 90u

#define COUNT 28

#define COUNTER 8

#define DELAY 10

#define GO 65536

unsigned notes[NOTES\_AMOUNT][3] = {

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent\*5},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent},

{a4, duration \* 3, indent},

{a3, duration \* 3, indent},

{f3, duration, indent},

{f3, duration, indent},

{e3, duration, indent},

{a3, duration, indent\*3},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{e4, duration, indent},

{A5, duration, indent},

{a5, duration, indent \* 3},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{e4, duration, indent},

{A5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{A5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{A5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{A5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{a5, duration, indent},

{A5, duration, indent},

};

void state\_words(void) {

unsigned channel, state;

// Port 40h (channel 0, system clock interruption)

// Port 41h (channel 1, memory regeneration)

// Port 42h (channel 2, speaker sound)

int ports[] = { 0x40, 0x41, 0x42 };

// 11 - RBC (always 11)

// 1 - not remember CE

// 0 - read chanel state

// 001, 010, 100 - chanel

// 0 - always 0

// 11 1 0 001 0, 11 1 0 010 0, 11 1 0 100 0

int control\_word[] = { 226, 228, 232 };

// Almost the same as control register (in set\_frequency)

// 6 - check is timer ready to read

// 7 - OUT: check out line state

char state\_word[] = "76000000";

int i;

printf("Status word: \n");

for (channel = 0; channel < 3; channel++) {

// Select channel (CLC commands)

outp(0x43, control\_word[channel]);

// Read state

state = inp(ports[channel]);

// Convert state into binary

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

state\_word[i] = (char)((state % 2) + '0');

state /= 2;

}

printf("Channel %d: %s\n", channel, state\_word);

}

}

void set\_frequency(unsigned divider) {

unsigned long kd = 1193180 / divider;

// 10 11 011 0:

// 10 - chanel

// 11 - read/write low, then high byte

// 011 - meander

// 0 - bin

outp(0x43, 0xB6);

// The smallest byte of the frequency divider

outp(0x42, kd % 256);

kd /= 256;

// The highest byte of the frequency divider

outp(0x42, kd);

}

void play\_music(void) {

int i;

for (i = 0; i < NOTES\_AMOUNT; i++) {

set\_frequency(notes[i][0]);

// Turn on speaker using first 2 bits:

// 0 - turn on/off chanel 2 in sys timer

// 1 - turn on/off dynamic

outp(0x61, inp(0x61) | 0x03);

delay(notes[i][1]);

// Turn off speaker

outp(0x61, inp(0x61) & 0xFC);

delay(notes[i][2]);

}

}

void div(void)

{

// port 40h (channel 0, system clock interrupion)

// port 41h (channel 1, memory regeneration)

// port 42h (channel 2, speaker sound)

int channel;

int ports[] = { 0x40, 0x41, 0x42 };

int controlWord[] = { 0x0, 0x40, 0x80 }; // CLC commands: for reading of current state of register counter of channel

unsigned byte, lowByte, highByte, maxByte;

printf("\nDivision factor: \n");

for (channel = 0; channel < 3; channel++)

{

byte = 0;

maxByte = 0;

for (unsigned long i = 0; i < GO; i++)

{

outp(0x43, controlWord[channel]); // select channel

lowByte = inp(ports[channel]); // read the smallest byte

highByte = inp(ports[channel]); // read the highest byte

byte = highByte \* 256 + lowByte; // generate byte

if (byte > maxByte)

maxByte = byte;

}

printf("\nChannel %d: %4X\n", channel, maxByte);

}

}

void sound(void)

{

int countHZ, byte;

int HZ[COUNTER] = { 329, 329, 329, 415, 523, 659, 587, 523 };

int MS[COUNTER] = { 200, 100, 200, 400, 200, 200, 400, 200 };

int countDelay[COUNTER] = { 100, 100, 100, 100, 100, 100, 100, 100 };

long unsigned base = 1193180; // IRQ 18.2 times per second

for (countHZ = 0; countHZ < COUNTER; countHZ++)

{

// 2th channel setting:

// port 42h (system timer, channel 2, speaker sound)

// port 43h (command register)

outp(0x43, 0xB6); // 0xB6 - configure 2 channel by port 43h

byte = base / HZ[countHZ];

outp(0x42, byte % 256); // (low) the smallest byte of the frequency divider

outp(0x42, byte /= 256); // (high) the highest byte of the frequency divider

outp(0x61, inp(0x61) | 3); // turn ON

delay(MS[countHZ]); // wait

outp(0x61, inp(0x61) & 0xFC); // turn OFF

delay(countDelay[countHZ]); // wait

}

}

int main(void) {

clrscr();

char command;

do

{

printf("\n----------------------------\n");

printf("1. Morgenshtern - Cadillac\n");

printf("2. Play a default sound\n");

printf("3. Division factor\n");

printf("4. Status word\n");

printf("0. Exit the program\n");

printf("----------------------------\n");

printf("\nSelect command: ");

fflush(stdin);

scanf("%s", &command);

switch (command)

{

case '1': play\_music(); break;

case '2': sound(); break;

case '3': div(); break;

case '4': state\_words(); break;

default: break;

}

} while (command != '0');

return 0;

}

1. Тестирование программы

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

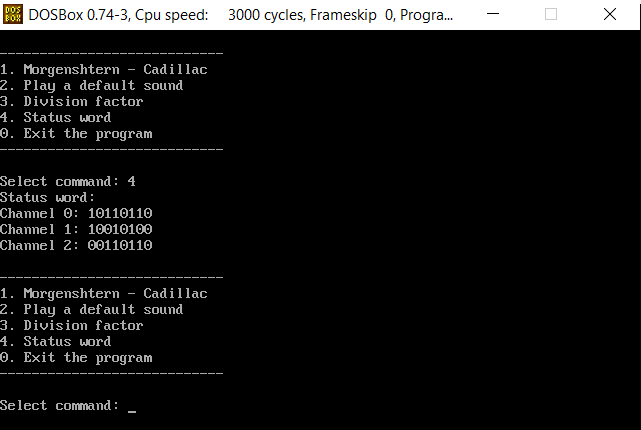


Рисунок 4.1 – Результат работы программы при выводе слов состояния каналов таймера.

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.