БГУИР

Кафедра ЭВМ

Отчет по лабораторной работе №4

Тема: «Часы реального времени»

Выполнил:

студент группы 750501 Новицкий А.Д.

Проверил:

к.т.н., доцент Одинец Д.Н.

Минск

2019

**СОДЕРЖАНИЕ**

1. Постановка задач………………………………………………………………..3

2. Алгоритм решения задачи……………………………………………………...4

3. Листинг программы…………………………………………………………..5-8

4. Заключение..…………………………………………………………………….9

**ПОСТАНОВКА ЗАДАЧ**

Под MS DOS написать программу, которая:

1) считывает и устанавливает время в часах реального времени;

а также по вариантам:

а) устанавливает будильник

б) реализует задержку с точностью до 1 миллисекунды

**АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ**

1. Перед установкой значений времени вызывается функция freeze(), которая считывает и анализирует старший байт регистра состояния 1 на предмет доступности значений для чтения и записи. Когда этот бит установлен в '0', отключается внутренний цикл обновления часов реального времени: для этого старший бит регистра состояния 2 устанавливается в '1'.
2. Считывание или запись значений времени происходит следующим образом: в порт 70h отправляется индекс регистра CMOS, соответствующий значению времени (секунды, часы и т. д.), затем происходит чтение значения из порта 71h (или запись значения в порт).
3. После установки значений времени вызывается функция unfreeze(), которая возобновляет внутренний цикл обновления часов реального времени.
4. Для реализации функции задержки заменён обработчик прерывания 0x70, в котором происходит отсчёт миллисекунд. Для включения периодического прерывания, происходящего примерно каждую миллисекунду, 6-й бит регистра B устанавливается в '1'.

**ЛИСТИНГ ПРОГРАММЫ**

#include <dos.h>

#include <ctype.h>

#include <stdio.h>

#include <conio.h>

#include <string.h>

#include <stdlib.h>

int milsec = 0;

void interrupt(\*oldI70h)(...);

void interrupt newI70h(...) {

outp(0x70, 0x0C);

inp(0x71);

outp(0x20, 0x20);

outp(0xA0, 0x20);

}

void interrupt(\*oldI4Ah)(...);

void interrupt newI4Ah(...) {

short value;

oldI4Ah();

outp(0x43, 0xB6);

value = 1193180 / 850;

outp(0x42, (unsigned char)value);

outp(0x42, (unsigned char)(value >> 8));

outp(0x61, (inp(0x61) | 3));

delay(1000);

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

outp(0x20, 0x20);

}

unsigned char get\_param(int reg)

{

unsigned char result;

outp(0x70, reg);

result = inp(0x71);

return result;

}

void wait() {

do {

outp(0x70, 0x0A);

} while (inp(0x71) & 0x80);

}

void disableUpdate() {

wait();

outp(0x70, 0x0B);

outp(0x71, (inp(0x71) | 0x80));

printf("Time update was disabled\n");

}

void enableUpdate() {

wait();

outp(0x70, 0x0B);

outp(0x71, (inp(0x71) & 0x7F));

printf("Time update was enabled\n");

}

int inputInt(int min, int max) {

int number;

do {

fflush(stdin);

} while (!scanf("%d", &number) || number<min || number>max);

return number;

}

int getInt(int BCD) {

return BCD % 16 + BCD / 16 \* 10;

}

unsigned char getBCDcode(int IntNumber) {

return (unsigned char)((IntNumber / 10) << 4) | (IntNumber % 10);

}

void set\_param(int value, int reg, int min, int max)

{

value = inputInt(min, max);

outp(0x70, reg);

outp(0x71, getBCDcode(value));

}

void getTotal()

{

char months[13][12] = { {"Unknown"},{"January"},{"February"},{"March"},{"April"},{"May"},{"June"},{"July"},{"August"},{"September"},{"October"},{"November"},{"December"} };

char weekdays[8][12] = { "Unknown","Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday" };

unsigned char hour, minute, second, year, month, day, weekday;

disableUpdate();

printf("The result of data and time:\n");

hour = get\_param(0x04);

minute = get\_param(0x02);

second = get\_param(0x00);

year = get\_param(0x09);

month = get\_param(0x08);

day = get\_param(0x07);

weekday = get\_param(0x06);

printf("Time: %d:%d:%d\nData: %s %d,20%d - %s\n", getInt(hour), getInt(minute), getInt(second), months[getInt(month)], getInt(day), getInt(year), weekdays[getInt(weekday)]);

enableUpdate();

}

void setTotal()

{

int value;

disableUpdate();

printf("\nHour: ");

set\_param(value, 0x04, 0, 24);

printf("\nMinutes: ");

set\_param(value, 0x02, 0, 60);

printf("\nSeconds: ");

set\_param(value, 0x00, 0, 60);

printf("\nDay of week: ");

set\_param(value, 0x06, 1, 7);

printf("\nDay of month: ");

set\_param(value, 0x07, 1, 31);

printf("\nMonth: ");

set\_param(value, 0x08, 1, 12);

enableUpdate();

printf("Time is successfully set\n");

}

void delayTime() {

unsigned long delayPeriod;

unsigned char value;

\_disable();

oldI70h = getvect(0x70);

setvect(0x70, newI70h);

\_enable();

printf("\nEnter delay time in milliseconds: ");

scanf("%ld", &delayPeriod);

printf("Delaying ...");

value = inp(0xA1) & 0xFE; // enabel interrupt RTC 11111110

outp(0xA1, value);

outp(0x70, 0x0B);

value = inp(0x71) | 0x40; // enable IRG8

outp(0x70, 0x0B);

outp(0x71, value);

milsec = 0;

while (milsec != delayPeriod);

printf("\nEnd delay of %d ms\n", milsec);

setvect(0x70, oldI70h);

enableUpdate();

}

void setAlarm()

{

int value;

disableUpdate();

printf("Enter hour: ");

value = inputInt(0, 24);

outp(0x70, 0x05);

outp(0x71, getBCDcode(value));

printf("Enter minute: ");

value = inputInt(0, 60);

outp(0x70, 0x03);

outp(0x71, getBCDcode(value));

printf("Enter second: ");

value = inputInt(0, 60);

outp(0x70, 0x01);

outp(0x71, getBCDcode(value));

outp(0x70, 0x0B);

outp(0x71, (inp(0x71) | 0x20));

enableUpdate();

printf("Alarm is set\n");

}

void viewAlarm()

{

unsigned char value;

disableUpdate();

outp(0x70, 0x05);

value = inp(0x71);

printf("Alarm - %d:", getInt(value));

outp(0x70, 0x03);

value = inp(0x71);

printf("%d:", getInt(value));

outp(0x70, 0x01);

value = inp(0x71);

printf("%d\n", getInt(value));

enableUpdate();

}

int main()

{

char choose, value;

oldI4Ah = getvect(0x4A);

setvect(0x4A, newI4Ah);

while (choose != '0') {

clrscr();

printf("\n1)Time info\n2)Set time\n3)Set alarm time\n4)Delay time(alarm clocks set)");

printf("\n5)Show alarm clocks\n0)Leave away\n");

choose = getch();

switch (choose) {

case '1': getTotal(); break;

case '2': setTotal(); break;

case '3': delayTime(); break;

case '4': setAlarm(); break;

case '5': viewAlarm(); break;

case '0': break;

}

getch();

}

setvect(0x4A, oldI4Ah);

return 0;

}

**ЗАКЛЮЧЕНИЕ**

Регистры часов реального времени:

0 - счётчик секунд

1 - регистр секунд будильника

2 - счётчик минту

3 - счётчик минут будильника

4 - счётчик часов

5 - счётчик часов будильника

6 - счётчик дней недели (1 - воскресенье)

7 - счётчик дней месяца

8 - счётчик месяцев

9 - счётчик лет (последние две цифры текущего года)

В регистрах состояния отражается:

* бит 7 - запрещено обновление часов (устанавливают перед запись новых значений в регистры даты и часов)
* бит 6 - вызов периодического прерывания (IRQ8)
* бит 5 - вызов прерывания при срабатывании будильника
* бит 4 - вызов прерывания по окончании обновления времени
* бит 3 - включена генерация прямоугольных импульсов
* бит 2 - 1/0 форматы даты и времени двоичный/BCD
* бит 1 - 1/0 24/12-часовой формат
* бит 0 - автоматический переход на летнее время в апреле и октябре

Через порты 70h и 71h происходит доступ к ЧРВ для записи и чтения/записи.