**Приложение А**

Исходный текст программы микроконтроллера

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This program was created by the

CodeWizardAVR V3.12 Advanced

Automatic Program Generator

© Copyright 1998-2014 Pavel Haiduc, HP InfoTech s.r.l.

http://www.hpinfotech.com

Project :

Version :

Date : 15.11.2017

Author :

Company :

Comments:

Chip type : AT90S8515

AVR Core Clock frequency: 8,000000 MHz

Memory model : Small

External RAM size : 0

Data Stack size : 128

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <90s8515.h>

// Alphanumeric LCD functions

#include <alcd.h>

#include <delay.h>

#include <stdio.h>

#include <eeprom.h> //подключаем EEPROM

// Declare your global variables here.

int sec=0,mins=0,hrs=0;

\_\_eeprom int ustr[8][4];

int loc\_time[4];

int skip=0;

char str[16];

// Прерывание таймера 1.

// Timer1 overflow interrupt service routine

interrupt [TIM1\_OVF] void timer1\_ovf\_isr(void)

{

int i=0;

// Устанавливаем регистры Т1 так, чтобы следующее прерывание снова возникло через //секунду

// Reinitialize Timer1 value

if (skip==0)

{

TCNT1H=0xBDC >> 8;

TCNT1L=0xBDC & 0xff;

}

else

{

TCNT1H=0xFFEE >> 8;

TCNT1L=0xFFEE & 0xff;

}

sec++; //прибавляем секунду

if (sec==60) //если секунд 60, то +1 минута

{

mins++;

sec=0;

if (mins==60) //если минут 60, то +1 час

{

hrs++;

mins=0;

if (hrs==24)

{

hrs=0;

}

}

}

while (i<8) //для всех устройств (от 0 до 7)

{

if (ustr[i][0]==1) //если устройство сейчас включено, то + 1 сек к работе

{

ustr[i][1]++;

if (ustr[i][1]==60)

{

ustr[i][1]=0;

ustr[i][2]++;

if (ustr[i][2]==60)

{

ustr[i][2]=0;

ustr[i][3]++;

}

}

}

i++;

}

}

void main(void)

{

// Declare your local variables here

int i=0;

// Input/Output Ports initialization

// Port A initialization

// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=Out Bit1=Out Bit0=Out

DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) | (0<<DDA2) | (0<<DDA1) | (0<<DDA0);

// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=0 Bit1=0 Bit0=0

PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) | (0<<PORTA2) | (0<<PORTA1) | (0<<PORTA0);

// Port B initialization

// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In

DDRB=(0<<DDB7) | (0<<DDB6) | (0<<DDB5) | (0<<DDB4) | (0<<DDB3) | (0<<DDB2) | (0<<DDB1) | (0<<DDB0);

// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T

PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) | (0<<PORTB4) | (0<<PORTB3) | (0<<PORTB2) | (0<<PORTB1) | (0<<PORTB0);

// Port C initialization

// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In

DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2) | (0<<DDC1) | (0<<DDC0);

// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T

PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) | (0<<PORTC0);

// Port D initialization

// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In

DDRD=(0<<DDD7) | (0<<DDD6) | (0<<DDD5) | (0<<DDD4) | (0<<DDD3) | (0<<DDD2) | (0<<DDD1) | (0<<DDD0);

// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T

PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) | (0<<PORTD4) | (0<<PORTD3) | (0<<PORTD2) | (0<<PORTD1) | (0<<PORTD0);

// Timer/Counter 0 initialization

// Clock source: System Clock

// Clock value: Timer 0 Stopped

TCCR0=(0<<CS02) | (0<<CS01) | (0<<CS00);

TCNT0=0x00;

//Инициализация таймера 1

// Timer/Counter 1 initialization

// Clock source: System Clock

// Clock value: 31,250 kHz

// Mode: Normal top=0xFFFF

// OC1A output: Disconnected

// OC1B output: Disconnected

// Noise Canceler: Off

// Input Capture on Falling Edge

// Timer Period: 1 s

// Timer1 Overflow Interrupt: On

// Input Capture Interrupt: Off

// Compare A Match Interrupt: Off

// Compare B Match Interrupt: Off

TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<PWM11) | (0<<PWM10);

TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<CTC1) | (1<<CS12) | (0<<CS11) | (0<<CS10);

TCNT1H=0x85;

TCNT1L=0xEE;

OCR1AH=0x00;

OCR1AL=0x00;

OCR1BH=0x00;

OCR1BL=0x00;

// Настройка прерывания таймера 1

// Timer(s)/Counter(s) Interrupt(s) initialization

TIMSK=(0<<TICIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (1<<TOIE1) | (0<<TOIE0);

// External Interrupt(s) initialization

// INT0: Off

// INT1: Off

GIMSK=(0<<INT1) | (0<<INT0);

MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);

// UART initialization (инициализация)

// Communication Parameters: 8 Data, 1 Stop, No Parity

// UART Receiver: Off

// UART Transmitter: On

// UART Baud Rate: 9600

UCR=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (1<<TXEN) | (0<<CHR9) | (0<<RXB8) | (0<<TXB8);

UBRR=0x33;

// Analog Comparator initialization

// Analog Comparator: Off

// The Analog Comparator's positive input is

// connected to the AIN0 pin

// The Analog Comparator's negative input is

// connected to the AIN1 pin

ACSR=(1<<ACD) | (0<<ACO) | (0<<ACI) | (0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);

// SPI initialization

// SPI disabled

SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SPR0);

// Alphanumeric LCD initialization (ЖК-дисплей - инициализация)

// Connections are specified in the

// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:

// RS - PORTC Bit 0

// RD - PORTC Bit 1

// EN - PORTC Bit 2

// D4 - PORTC Bit 3

// D5 - PORTC Bit 4

// D6 - PORTC Bit 5

// D7 - PORTC Bit 6

// Characters/line: 16

lcd\_init(16);

// Global enable interrupts

//Разрешение глобальных прерываний (чтобы контроллер воспринимал и обрабатывал //прерывания и прерывания Т1 работали)

#asm("sei")

loc\_time[3]=0;

while (i<8) //инициализируем переменные, в которых будем хранить время работы устройств (выставляем их в 0)

{

ustr[i][0]=0;

ustr[i][1]=0;

ustr[i][2]=0;

ustr[i][3]=0;

i++;

}

while (1) //основной цикл программы

{

switch (loc\_time[3]) //смотрим на переменную, которая выводится на экран. В зависимости от первого числа в ней, пишем в нее текущее время счетчика, либо же время одного из устройств

{

case 0:

loc\_time[0]=hrs;

loc\_time[1]=mins;

loc\_time[2]=sec;

break;

case 1:

loc\_time[0]=ustr[0][3];

loc\_time[1]=ustr[0][2];

loc\_time[2]=ustr[0][1];

break;

case 2:

loc\_time[0]=ustr[1][3];

loc\_time[1]=ustr[1][2];

loc\_time[2]=ustr[1][1];

break;

case 3:

loc\_time[0]=ustr[2][3];

loc\_time[1]=ustr[2][2];

loc\_time[2]=ustr[2][1];

break;

case 4:

loc\_time[0]=ustr[3][3];

loc\_time[1]=ustr[3][2];

loc\_time[2]=ustr[3][1];

break;

case 5:

loc\_time[0]=ustr[4][3];

loc\_time[1]=ustr[4][2];

loc\_time[2]=ustr[4][1];

break;

case 6:

loc\_time[0]=ustr[5][3];

loc\_time[1]=ustr[5][2];

loc\_time[2]=ustr[5][1];

break;

case 7:

loc\_time[0]=ustr[6][3];

loc\_time[1]=ustr[6][2];

loc\_time[2]=ustr[6][1];

break;

case 8:

loc\_time[0]=ustr[7][3];

loc\_time[1]=ustr[7][2];

loc\_time[2]=ustr[7][1];

break;

}

//если 0 часов или 12 часов, то выполняем пересылку данных в ПЭВМ

if (((hrs==0) & (mins==0) & (sec==0)) || ((hrs==12) & (mins==0) & (sec==0)))

{

char udr\_buf[12]; // темповая строка для отправки в пэвм

int j=0;

i=0;

while (i<8) //для каждого из внешних устройств

{

sprintf(udr\_buf,"%d:%d:%d ",ustr[i][3],ustr[i][2],ustr[i][1]);

// пишем в строку время работы устройства

while (j<6) //для каждого символа в строке

{

UDR=udr\_buf[j]; //отправляем символ в ПЭВМ

delay\_ms(5); // ждём 5 мс

j++; // переходим к следующему символу

}

UDR=' '; // отправляем пробел, чтобы разделить время работы устройств

j=0;

sprintf(udr\_buf,"%s"," ");// очищаем строку

i++; // переходим к следующему устройству

}

i=0;

delay\_ms(900);

}

if (PIND.4==0) // кнопка ускорения времени!

{

while (PIND.4==0)

delay\_ms(5);

if (skip==0)

skip=1;

else

skip=0;

} //Она выставляет "skip" в 1, который считывается в прерывании и выставляет биты //таймера так,

// что следующее прерывание происхзодит не через секунду, а через долю секунды, //тем самым ускоряет время

if (PINA.0==0) // устранение дребезга

{

while(PINA.0==0)

delay\_ms(5);

if (loc\_time[3]==0) // если сейчас на дисплее - просто таймер, переход к устройствам

{

loc\_time[3]=1;

lcd\_clear();

lcd\_gotoxy(10,0);

sprintf(str,"%d",loc\_time[3]);

lcd\_puts(str);

}

else //если в переменной loc\_time устройства, переход обратно к отображению //таймера (текущего времени)

{

lcd\_clear();

loc\_time[3]=0;

}

}

if (loc\_time[3]!=0) // если в переменной loc\_time устройства, можно переключать их //номер кнопками А.1 и А.2

{

if (PINA.1==0)

{

while(PINA.1==0) // устранение дребезга

delay\_ms(5);

if (loc\_time[3]<8)

loc\_time[3]+=1;

lcd\_clear();

lcd\_gotoxy(10,0);

sprintf(str,"%d",loc\_time[3]);

lcd\_puts(str);

}

if (PINA.2==0) // устранение дребезга

{

while(PINA.2==0)

delay\_ms(5);

if (loc\_time[3]>1)

loc\_time[3]-=1;

lcd\_clear();

lcd\_gotoxy(10,0);

sprintf(str,"%d",loc\_time[3]);

lcd\_puts(str);

}

}

// вывод на экран переменной loc\_time (т.е. или таймера, или время работы одного из //устройств)

lcd\_gotoxy(0,0);

lcd\_puts("Time:");

lcd\_gotoxy(0,1);

sprintf(str,"%d:%d:%d ",loc\_time[0],loc\_time[1],loc\_time[2]);

lcd\_puts(str);

//проверка PIN-ов, на которые приходят сигналы с ВУ и сброс или установка //соответствующих переменных (1 - устройство работает, 0 - не работает)

if (PINB.0==0)

ustr[0][0]=1;

else

ustr[0][0]=0;

if (PINB.1==0)

ustr[1][0]=1;

else

ustr[1][0]=0;

if (PINB.2==0)

ustr[2][0]=1;

else

ustr[2][0]=0;

if (PINB.3==0)

ustr[3][0]=1;

else

ustr[3][0]=0;

if (PINB.4==0)

ustr[4][0]=1;

else

ustr[4][0]=0;

if (PINB.5==0)

ustr[5][0]=1;

else

ustr[5][0]=0;

if (PINB.6==0)

ustr[6][0]=1;

else

ustr[6][0]=0;

if (PINB.7==0)

ustr[7][0]=1;

else

ustr[7][0]=0;

}

}