Coding Mikro Controller

Monitoring\_suhu\_uart.c

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

This program was produced by the

CodeWizardAVR V2.05.3 Standard

Automatic Program Generator

© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.

http://www.hpinfotech.com

Project :

Version :

Date : 6/28/2019

Author : unknown

Company :

Comments:

Chip type : ATmega8535

Program type : Application

AVR Core Clock frequency: 11.059200 MHz

Memory model : Small

External RAM size : 0

Data Stack size : 128

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

#include <mega8535.h>

// Standard Input/Output functions

#include <stdlib.h>

#include <stdio.h>

#include <delay.h>

#include "myLCD.h"

#include "myThermo.h"

// Declare your global variables here

char stringbuffer[16], temp[6];

void main(void)

{

// Declare your local variables here

// Input/Output Ports initialization

// Port A initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTA=0x00;

DDRA=0x00;

// Port B initialization

// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out

// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0

PORTB=0x00;

DDRB=0xFF;

// Port C initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In

// State7=P State6=P State5=P State4=P State3=P State2=P State1=P State0=P

PORTC=0xFF;

DDRC=0x00;

// Port D initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=Out Func0=Out

// State7=T State6=T State5=T State4=T State3=T State2=P State1=0 State0=0

PORTD=0x04;

DDRD=0x0C;

// Timer/Counter 0 initialization

// Clock source: System Clock

// Clock value: Timer 0 Stopped

// Mode: Normal top=0xFF

// OC0 output: Disconnected

TCCR0=0x00;

TCNT0=0x00;

OCR0=0x00;

// Timer/Counter 1 initialization

// Clock source: System Clock

// Clock value: Timer1 Stopped

// Mode: Normal top=0xFFFF

// OC1A output: Discon.

// OC1B output: Discon.

// Noise Canceler: Off

// Input Capture on Falling Edge

// Timer1 Overflow Interrupt: Off

// Input Capture Interrupt: Off

// Compare A Match Interrupt: Off

// Compare B Match Interrupt: Off

TCCR1A=0x00;

TCCR1B=0x00;

TCNT1H=0x00;

TCNT1L=0x00;

ICR1H=0x00;

ICR1L=0x00;

OCR1AH=0x00;

OCR1AL=0x00;

OCR1BH=0x00;

OCR1BL=0x00;

// Timer/Counter 2 initialization

// Clock source: System Clock

// Clock value: Timer2 Stopped

// Mode: Normal top=0xFF

// OC2 output: Disconnected

ASSR=0x00;

TCCR2=0x00;

TCNT2=0x00;

OCR2=0x00;

// External Interrupt(s) initialization

// INT0: Off

// INT1: Off

// INT2: Off

MCUCR=0x00;

MCUCSR=0x00;

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

TIMSK=0x00;

// USART initialization

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

// USART Receiver: Off

// USART Transmitter: On

// USART Mode: Asynchronous

// USART Baud Rate: 9600

UCSRA=0x00;

UCSRB=0x08;

UCSRC=0x86;

UBRRH=0x00;

UBRRL=0x47;

// Analog Comparator initialization

// Analog Comparator: Off

// Analog Comparator Input Capture by Timer/Counter 1: Off

ACSR=0x80;

SFIOR=0x00;

// ADC initialization

// ADC disabled

ADCSRA=0x00;

// SPI initialization

// SPI disabled

SPCR=0x00;

// TWI initialization

// TWI disabled

TWCR=0x00;

LCD\_Init();

while (1)

{

// Place your code here

if(PINC.0 == 0){

ftoa(readCelcius(), 2, temp);

sprintf(stringbuffer, "suhu: %s'C ", temp);

goto proses;

}

if(PINC.1 == 0){

ftoa(readFahrenheit(), 2, temp);

sprintf(stringbuffer, "suhu: %s'F ", temp);

goto proses;

}

if(PINC.2 == 0){

ftoa(readKelvin(), 2, temp);

sprintf(stringbuffer, "suhu: %s'K ", temp);

goto proses;

}

sprintf(stringbuffer, ">pilih tampilan ");

proses:

LCD\_Tulis(0, "Monitor Suhu");

LCD\_Tulis(1, stringbuffer);

stringbuffer[15] = '\r';

puts(stringbuffer);

delay\_ms(1000);

}

}

myThermo.h

#define csPin PORTD.2

#define sckPin PORTD.3

#define soPin PIND.4

unsigned char read\_spi(){

int i;

unsigned char datain = 0;

LCD\_GotoXY(0,0);

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

sckPin = 0;

delay\_ms(1);

if(soPin){

datain = datain | (1 << i);

}

sckPin = 1;

delay\_ms(1);

}

return datain;

}

float readCelcius(){

uint16\_t v;

csPin = 0;

delay\_ms(1);

v = read\_spi();

v = v << 8;

v = v | read\_spi();

csPin = 1;

if(v & 0x4){

return 999.999;

}

v = v >> 3;

return v\*0.25;

}

float readFahrenheit(){

return readCelcius() \* 9.0/5.0 + 32;

}

float readKelvin(){

return readCelcius() + 273;

}

myLCD.h

// Deklarasi PIN MCU

#define LCD\_EN PORTB.2

#define LCD\_RS PORTB.3

#define LCD\_PORT PORTB

// deklarasi tipe data (jika sudah pernah bisa dihapus)

#define int8\_t signed char

#define uint8\_t unsigned char

#define int16\_t signed short

#define uint16\_t unsigned short

#define DelayLCD 10 // delay animasi dalam mili second

#pragma used+

// Menuliskan init command pada LCD

void LCD\_Init\_Cmd(uint8\_t xData)

{ LCD\_PORT = 0B00000100 | (xData & 0xF0);

delay\_us(1); LCD\_EN = 0;

}

// Penulisan command (secara umum) ke LCD

void LCD\_Perintah(uint8\_t xData)

{ //LCD\_RS=0; LCD\_RW=0; LCD\_EN=1;

LCD\_RS=0;

LCD\_PORT = 0B00000100 | (xData & 0xF0);

delay\_us(1); LCD\_EN = 0;

delay\_us(5);

LCD\_PORT = 0B00000100 | (xData << 4);

delay\_us(1); LCD\_EN = 0;

delay\_us(700);

LCD\_EN = 1;

}

// Menuliskan data ke LCD

void LCD\_Data(uint8\_t xData)

{ //LCD\_RS=1; LCD\_RW=0; LCD\_EN=1;

LCD\_RS=1;

LCD\_PORT = 0B00001100 | (xData & 0xF0);

delay\_us(1); LCD\_EN = 0;

delay\_us(5);

LCD\_PORT = 0B00001100 | (xData << 4);

delay\_us(1); LCD\_EN = 0;

delay\_us(40);

LCD\_EN = 1;

}

// Inisialisasi LCD (mode 4 bit data)

void LCD\_Init(void)

{

delay\_ms(50); LCD\_Init\_Cmd(0x30);

delay\_ms(5); LCD\_Init\_Cmd(0x30);

delay\_ms(1); LCD\_Init\_Cmd(0x30);

delay\_ms(1); LCD\_Init\_Cmd(0x20);

LCD\_Perintah(0x28); //4-bit/2-line

LCD\_Perintah(0x10); // Set cursor

LCD\_Perintah(0x0c); // Display ON; Cursor off

LCD\_Perintah(0x06); // Entry mode =increment, no shift

LCD\_Perintah(0x01); // Hapus layer

delay\_ms(1);

}

// Program penulisan pengaturan cursor LCD

void LCD\_GotoXY(uint8\_t x, uint8\_t y)

{ uint8\_t baris;

if (y==0) LCD\_Perintah(0x80 + x); // baris 1 --> y=0

else LCD\_Perintah(0x80 + x + 0x40); // baris 2 --> y=1

}

// Program penulisan text pada LCD dari flash

void LCD\_TextF(uint8\_t flash \*text) // menuliskan string ke LCD

{ while (\*text!=0)

{ LCD\_Data(\*text);text++;}

}

// Program penulisan text pada LCD dari variabel

void LCD\_Text(uint8\_t \*text) // menuliskan string ke LCD

{ while (\*text!=0)

{ LCD\_Data(\*text);text++;}

}

// Program penulisan text pada LCD dengan pemilihan baris

void LCD\_TulisF(uint8\_t Baris, uint8\_t flash \*text) // menuliskan string ke LCD

{ LCD\_GotoXY(0,Baris);LCD\_TextF(text);

}

// Program penulisan text pada LCD dengan pemilihan baris

void LCD\_Tulis(uint8\_t Baris, uint8\_t \*text) // menuliskan string ke LCD

{ LCD\_GotoXY(0,Baris);LCD\_Text(text);

}

// Program menghapus layar LCD

void LCD\_Hapus(void) // menghapus seluruh layar

{ LCD\_Perintah(0x01);

delay\_ms(1);

}

// Program menghapus data pada baris LCD

void LCD\_HapusBaris(uint8\_t Baris) // menghapus baris tabpa delay

{ uint8\_t i;

LCD\_GotoXY(0,Baris);

for(i=0;i<16;i++) LCD\_Data(' ');

}

// Program menghapus data pada baris LCD dari sebelah kiri

void LCD\_HapusKiri(uint8\_t Baris) // menghapus layar dari arah kiri dgn delay pada(0=baris 1, 1=Baris 2)

{ uint8\_t i;

LCD\_GotoXY(0,Baris);

for(i=0;i<16;i++) { LCD\_Data(' '); delay\_ms(DelayLCD);}

}

// Program menghapus data pada baris LCD dari sebelah kanan

void LCD\_HapusKanan(uint8\_t Baris) // menghapus layar dari arah kanan dgn delay pada(0=baris 1, 1=Baris 2)

{ int8\_t i,j;

j = 0x80 + 0x40\*Baris;

for(i=15;i>-1;i--)

{ LCD\_Perintah (j+i); LCD\_Data(' '); delay\_ms(DelayLCD);}

}

// Program menulis data pada baris LCD dari sebelah kiri

void LCD\_TulisKiri(uint8\_t Baris, uint8\_t flash \*text)

// menuliskan string dari arah kiri dgn delay pada (0=baris 1, 1=Baris 2)

{ uint8\_t i;

LCD\_GotoXY(0,Baris);

for(i=0;i<16;i++)

{ LCD\_Data(\*(text+i)); delay\_ms(DelayLCD);}

}

// Program menulis data pada baris LCD dari sebelah kanan

void LCD\_TulisKanan(uint8\_t Baris, uint8\_t flash \*text)

// menuliskan string dari arah kanan dgn delay

{ int8\_t i,j;

j = 0x80 + 0x40\*Baris;

for(i=15;i>-1;i--)

{ LCD\_Perintah (j+i); LCD\_Data(\*(text+i)); delay\_ms(DelayLCD);}

}

// Program menulis data pada baris LCD dari tengah

void LCD\_TulisTengah(uint8\_t Baris, uint8\_t flash \*text)

// menuliskan string dari arah kanan dgn delay

{ int8\_t i,j;

j = 0x80 + 0x40\*Baris;

for(i=7;i>-1;i--)

{ LCD\_Perintah (j+i); LCD\_Data(\*(text+i)); delay\_ms(DelayLCD);

LCD\_Perintah (j-i+15); LCD\_Data(\*(text-i+15)); delay\_ms(DelayLCD);

}

}

// Program menulis data pada baris LCD dari sebelah pinggir

void LCD\_TulisPinggir(uint8\_t Baris, uint8\_t flash \*text)

// menuliskan string dari arah kanan dgn delay

{ int8\_t i,j;

j = 0x80 + 0x40\*Baris;

for(i=0;i<8;i++)

{ LCD\_Perintah (j+i); LCD\_Data(\*(text+i)); delay\_ms(DelayLCD);

LCD\_Perintah (j-i+15); LCD\_Data(\*(text-i+15)); delay\_ms(DelayLCD);

}

}

// Program menulis data angka pada posisi kursor LCD (lebar 4 angka desimal)

void LCD\_Angka4(int16\_t x)

{ if(x<0){ x\*=-1; LCD\_Data('-');}

LCD\_Data(x/1000+0x30); // menulis ribuan

LCD\_Data((x%1000)/100+0x30); // menulis ratusan

LCD\_Data((x%100)/10+0x30); // menulis puluhan

LCD\_Data(x%10+0x30); // menulis satuan

}

// Program menulis data angka pada posisi kursor LCD (lebar 3 angka desimal)

void LCD\_Angka3(int16\_t x)

{ if(x<0){ x\*=-1; LCD\_Data('-');}

LCD\_Data(x/100+0x30); // menulis ratusan

LCD\_Data((x%100)/10+0x30); // menulis puluhan

LCD\_Data(x%10+0x30); // menulis satuan

}

// Program menulis data angka (byte) pada posisi kursor LCD (signed)

void LCD\_sByte(int8\_t x)

{ if(x<0){ x\*=-1; LCD\_Data('-');}

LCD\_Data(x/100+0x30); // menulis ratusan

LCD\_Data((x%100)/10+0x30); // menulis puluhan

LCD\_Data(x%10+0x30); // menulis satuan

}

// Program menulis data angka (byte) pada posisi kursor LCD (unsigned)

void LCD\_uByte(uint8\_t x)

{ LCD\_Data(x/100+0x30); // menulis ratusan

LCD\_Data((x%100)/10+0x30); // menulis puluhan

LCD\_Data(x%10+0x30); // menulis satuan

}

#pragma used-

Program GUI

Form1.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO.Ports;

using System.Diagnostics;

namespace Monitoring\_Suhu

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

foreach (string serial in SerialPort.GetPortNames())

{

comboBox1.Items.Add(serial);

}

}

private void button2\_Click(object sender, EventArgs e)

{

comboBox1.Items.Clear();

comboBox1.Text = "";

foreach (string serial in SerialPort.GetPortNames())

{

comboBox1.Items.Add(serial);

}

}

private void button1\_Click(object sender, EventArgs e)

{

if (button1.Text == "Connect")

{

try

{

if (comboBox1.Text.ToString() == "")

{

MessageBox.Show("Pilih comport terlebih dahulu");

return;

}

button1.Text = "Disconnect";

serialPort1.BaudRate = 9600;

serialPort1.PortName = comboBox1.Text.ToString();

serialPort1.Open();

serialPort1.DiscardInBuffer();

}

catch (System.IO.IOException ioe)

{

MessageBox.Show(ioe.Message);

}

}

else

{

try

{

serialPort1.Dispose();

serialPort1.Close();

button1.Text = "Connect";

}

catch (Exception ioe)

{

MessageBox.Show(ioe.Message);

}

}

}

private void serialPort1\_DataReceived(object sender, SerialDataReceivedEventArgs e)

{

double suhu = 0;

string tipe = "";

int start = 0;

int length = 1;

SerialPort sp = (SerialPort)sender;

try

{

string bufferedText = sp.ReadTo("\r\n");

Debug.WriteLine(bufferedText);

if (bufferedText.Contains('>') || !bufferedText.Contains("suhu"))

{

Debug.WriteLine("Jump");

goto jump;

}

start = bufferedText.IndexOf(':') + 1;

length = bufferedText.IndexOf('\'') - start;

suhu = double.Parse(bufferedText.Substring(start, length), System.Globalization.CultureInfo.InvariantCulture);

tipe = bufferedText.Substring(bufferedText.IndexOf('\'') + 1, 1);

plotData(suhu, tipe);

updateLbl(lblSuhu, suhu.ToString());

updateLbl(lblDateTime, DateTime.Now.ToString());

switch (tipe)

{

case "C":

updateLbl(lblSatuan, "Celcius");

break;

case "F":

updateLbl(lblSatuan, "Fahrenheit");

break;

case "K":

updateLbl(lblSatuan, "Kelvin");

break;

}

jump:

Debug.WriteLine(tipe);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

static string ttipe = "";

private void plotData(double suhu, string tipe)

{

BeginInvoke((Action)(() =>

{

if (ttipe != tipe)

{

ttipe = tipe;

chart1.Series[0].Points.Clear();

}

if (chart1.Series[0].Points.Count > 100)

{

chart1.Series[0].Points.RemoveAt(0);

}

chart1.Series[0].Points.AddXY(DateTime.Now.ToLongTimeString(), suhu);

chart1.Refresh();

}));

}

private void updateLbl(System.Windows.Forms.Label lbl,string s)

{

if (lbl.InvokeRequired)

{

lbl.BeginInvoke((MethodInvoker)delegate () { lbl.Text = s; });

}

else

{

lbl.Text = s;

}

}

}

}

