#include <Mega32.h>

#include <delay.h>

#asm

.equ \_\_lcd\_port = 0x15

#endasm

#include <math.h>

#include <lcd.h>

#include <stdio.h>

#include <stdlib.h>

//define states for motor control

#define startState 0

#define inflate1State 1

#define inflate2State 2

#define deflateState 3

#define displayState 4

#define resetState 5

//define states for Measure control

#define Sys\_Measure 6

#define Sys\_Cal 7

#define Rate\_Measure 8

#define dias\_Measure 9

#define dias\_Cal 10

#define LCDwidth 16

void initialize(void);

//declare functions for motor control

void start\_state(void);

void inflate1\_state(void);

void inflate2\_state(void);

void deflatestate(void);

void display\_state(void);

void reset\_state(void);

//declare all functions for measuring control

void pressuremeasure(void);

void sysmeasure(void);

void syscal(void);

void ratemeasure(void);

void diasmeasure(void);

void diascal(void);

//declare variable for motor controls

unsigned char Maybe0,Maybe1,Maybe2,countlcd;

unsigned char currentState;

unsigned int timepress0, timepress1, timepress2, timelcd;

char lcd\_output[17];

//declare variable for measuring and calculating value

float DC\_gain;

unsigned char meas\_state;

unsigned int timing, timerate, timerun\_dias, timecount, timedeflate, timedisplay;

float maxpressure, pressure,accum\_data, press\_data;

unsigned char count, stop\_count;

//ADC data variabls

float Vref;

unsigned char data;

float adc\_data, former;

//define counter

unsigned char sys\_count,count\_average, countpulse;

//declare rate measure variable

float time\_pulse,pulse\_period, total\_pulse\_period, pulse\_per\_min;

//declare systolic and diastolic variable

float systolic, diastolic;

//declare all the threshold values

float TH\_sys, TH\_rate, TH\_dias;

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

//timer 0 compare ISR

interrupt [TIM0\_COMP] void timer0\_compare(void)

{

if(~PINB & 0x01) timepress0++;

if(~PINB & 0x02) timepress1++;

if(~PINB & 0x04) timepress2++;

timecount++;

timedeflate++;

//Decrement each time tast if they are not already zero

//timing for sampling data at every 40 msec

if(timing>0) --timing;

//-----------------------------------------------------

//run time for different tasks

//run timerate for measuring heart rate

if(timerate<6000) ++timerate;

//run timerun\_dias

if(timerun\_dias<2000) ++timerun\_dias;

//if(countlcd) timelcd++;

//run time for the display

if(timedisplay<2000) ++timedisplay;

}

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

// ADC Interrupt

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

interrupt [ADC\_INT] void adc\_complete(void)

{

data = ADCH;

//then calculate adc\_data into float;

adc\_data = (float)(((float)data)/256\*Vref);

//if signal is above threshold, go to calculate systolic pressure

if(meas\_state ==Sys\_Measure)

{

if(former<=TH\_sys && adc\_data>TH\_sys)

sys\_count++;

former = adc\_data;

}

//-----------------------------------------------------------

else if(meas\_state==Sys\_Cal)

{

if(count<4)

{

accum\_data=accum\_data+adc\_data;

count++;

}

if(count==4)

{

press\_data=accum\_data/4;

systolic = (press\_data/DC\_gain)\*9375;//calculate from adc\_data

meas\_state = Rate\_Measure;

countpulse=0;

former = 2.4; //set the initial point for rate measuring

count\_average=0;

}

}

//----------------------------------------------------------

else if(meas\_state==Rate\_Measure)

{

if(count\_average<5)

{

if(former<TH\_rate && adc\_data>TH\_rate && countpulse==0)

{

timerate=0;

countpulse=1;

former=adc\_data;

}

if(former<TH\_rate && adc\_data>TH\_rate && countpulse==1)

{

total\_pulse\_period=total\_pulse\_period+timerate;

timerate=0;

count\_average++; //finish reading one period

}

}//count\_average

former=adc\_data;

}// else if(meas\_state=Rate\_Measure)

//-------------------------------------------------------------

else if(meas\_state==dias\_Measure)

{

if(timerun\_dias<2000)

{

if(adc\_data>TH\_dias)

{ timerun\_dias=0; //reset time if the signal

//is still greater than threshold (so it will never reach 1999)

//if it doesn't reset,the time will stuck at 1999

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Dias measure");

}

}

if(timerun\_dias>=2000)

{

meas\_state = dias\_Cal;//if done go back to Sys\_Measure to be ready for next opt

}

}

//-------------------------------------------------------------

else if(meas\_state==dias\_Cal)

{

diastolic = (adc\_data/DC\_gain)\*9375;//calculate from adc\_data

meas\_state = Sys\_Measure;

currentState = displayState;

//open valve

PORTD=0;

}

timing = 40;//set time for another conversion

}// end of ADC interrupt

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

void main(void)

{

initialize();

while(1)

{

switch(currentState)

{

case startState:

start\_state();

break;

case inflate1State:

inflate1\_state();

break;

case inflate2State:

inflate2\_state();

break;

case deflateState:

deflatestate();

break;

case displayState:

display\_state();

break;

case resetState:

reset\_state();

break;

}

}

}

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

void start\_state(void)

{ sys\_count=0;

pressure = 0;

accum\_data=0;

press\_data=0;

count=0;

stop\_count=0;

maxpressure = 160;

meas\_state = Sys\_Measure;

former=TH\_sys-0.01;

timerun\_dias=0;

time\_pulse=0;

timerate=0;

timing=40;

total\_pulse\_period=0;

systolic=0;

diastolic=0;

pulse\_per\_min=0;

sys\_count=0;

count\_average=0;

countpulse=0;

if((~PINB & 0x01) && (timepress0 > 30)) Maybe0 = 1;

if(Maybe0 && (PINB == 0xff))

{

countlcd = 1;

timelcd = 0;

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Inflating");

currentState = inflate1State;

Maybe0 = 0;

timepress0 = 0;

timecount=0;

//turn on motor and close the valve

PORTD=0x03;

//activate ADC

}

}

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

void inflate1\_state(void)

{

if(timecount>=200)

{

timecount=0;

sprintf(lcd\_output,"%-i",(int)pressure);

lcd\_gotoxy(0,1);

lcd\_puts(lcd\_output);

}

if((~PINB & 0x02) && (timepress1 > 30)) Maybe1 = 1;

if(Maybe1 && (PINB == 0xff))

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Emergency Stop");

sprintf(lcd\_output,"%-i",(int)pressure);

lcd\_gotoxy(0,1);

lcd\_puts(lcd\_output);

//turn off motor and open the valve

PORTD=0;

currentState = resetState;

Maybe1 = 0;

timepress1 = 0;

countlcd = 0;

}

else

{

currentState = inflate2State;

}

}

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

void inflate2\_state(void)

{

ADMUX=0b00100001;//choose ADC1 for reading DC

//enable ADC and set prescaler to 1/128\*16MHz=125,000

//and uncheck interupt enable

//and start a conversion

ADCSR = 0b11000111;

data= ADCH;

adc\_data = (float)(((float)data)/256\*Vref);

pressure= (adc\_data/DC\_gain)\*9375;

if(pressure>=maxpressure) stop\_count++;

else stop\_count = 0;

if(stop\_count>=5)

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Deflating");

sprintf(lcd\_output,"%-i",(int)pressure);

lcd\_gotoxy(0,1);

lcd\_puts(lcd\_output);

//turn off motor but keep the valve

PORTD = 0x02;

delay\_ms(1000);

currentState = deflateState;

timedeflate = 0;

sprintf(lcd\_output,"%-i",(int)pressure);

lcd\_gotoxy(0,1);

lcd\_puts(lcd\_output);

}

else

{

currentState = inflate1State;

}

}

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

void deflatestate(void)

{

/\*if(timedeflate >= 1900)

{

PORTD = 0;

if(timedeflate >= 2000)

{

PORTD = 0x02;

timedeflate = 0;

}

}\*/

if((~PINB & 0x02) && (timepress1 > 30)) Maybe1 = 1;

if(Maybe1 && (PINB == 0xff))

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Emergency Stop");

sprintf(lcd\_output,"%-i",(int)pressure);

lcd\_gotoxy(0,1);

lcd\_puts(lcd\_output);

//turn off motor and open the valve

PORTD=0;

currentState = resetState;

Maybe1 = 0;

timepress1 = 0;

}

//if(done) --> Display state

if(currentState==deflateState) pressuremeasure(); //if still deflating, measure everything

}

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

void display\_state(void)

{

if(timedisplay<=1000)

{

if(timecount>=200)

{

lcd\_clear();

timecount=0;

lcd\_gotoxy(0,0);

lcd\_putsf("Sys");

lcd\_gotoxy(7,0);

lcd\_putsf("Dias");

lcd\_gotoxy(15,0);

lcd\_putsf("HR");

sprintf(lcd\_output,"%-i",(int)systolic);

lcd\_gotoxy(0,1);

lcd\_puts(lcd\_output);

sprintf(lcd\_output,"%-i",(int)diastolic);

lcd\_gotoxy(7,1);

lcd\_puts(lcd\_output);

sprintf(lcd\_output,"%-i",(int)pulse\_per\_min);

lcd\_gotoxy(14,1);

lcd\_puts(lcd\_output);

}

}

else if (timedisplay>1000&&timedisplay<2000)

{

if(timecount>=200)

{ lcd\_clear();

timecount=0;

lcd\_gotoxy(0,0);

lcd\_putsf("Black: Resume");

}

}

else

{

timedisplay=0;

}

if((~PINB & 0x04) && (timepress2 > 30)) Maybe2 = 1;

if(Maybe2 && (PINB == 0xff))

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("White: Start");

lcd\_gotoxy(0,1);

lcd\_putsf("Grey: Stop");

currentState = startState;

timepress2 = 0;

Maybe2=0;

systolic=0;

diastolic=0;

pulse\_per\_min=0;

}

}

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

void reset\_state(void)

{

if(timedisplay<=1000)

{

if(timecount>=200)

{ timecount=0;

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Emergency Stop");

}

}

else if (timedisplay>1000&&timedisplay<2000)

{

if(timecount>=200)

{ lcd\_clear();

timecount=0;

lcd\_gotoxy(0,0);

lcd\_putsf("Black: Resume");

}

}

else

{

timedisplay=0;

}

if((~PINB & 0x04) && (timepress2 > 30)) Maybe2 = 1;

if(Maybe2 && (PINB == 0xff))

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("White: Start");

lcd\_gotoxy(0,1);

lcd\_putsf("Grey: Stop");

currentState = startState;

timepress2 = 0;

Maybe2=0;

}

}

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

// Function to measure everything

//------------------------------------------------

void pressuremeasure(void)

{

switch (meas\_state)

{

case Sys\_Measure:

if(timing==0) sysmeasure(); //sampling signal at 40msec

break;

case Sys\_Cal:

if(timing==0) syscal();

break;

case Rate\_Measure:

if(timing==0) ratemeasure();

break;

case dias\_Measure:

diasmeasure();

break;

case dias\_Cal:

diascal();

break;

} //switch

}//pressuremeasure

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

void sysmeasure(void)

{

if(timing==0)

{ADMUX = 0b00100000; //choose ADC0 for reading AC

//enable ADC and set prescaler to 1/128\*16MHz=125,000

//and set interupt enable

//and start a conversion

ADCSR = 0b11001111;

}

if(sys\_count>=6)

{

meas\_state = Sys\_Cal;

timecount=0;

}

if(timecount>=200)

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Measuring");

timecount=0;

}

}

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

//this function is to calculate systolic pressure

void syscal(void)

{

ADMUX=0b00100001;//choose ADC1 for reading DC

//enable ADC and set prescaler to 1/128\*16MHz=125,000

//and set interupt enable

//and start a conversion

ADCSR = 0b11001111;

if(timecount>=200)

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Sys Cal");

timecount=0;

}

}//syscal

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

void ratemeasure(void)

{

ADMUX=0b00100000; //choose ADC0 for reading AC

//enable ADC and set prescaler to 1/128\*16MHz=125,000

//and set interupt enable

//and start a conversion

ADCSR = 0b11001111;

//calculate the mean of pulse rate

if(count\_average==5)

{

pulse\_period = total\_pulse\_period/5000;

pulse\_per\_min= 60/pulse\_period;

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Pulse Rate");

sprintf(lcd\_output,"%-i",(int)pulse\_per\_min);

lcd\_gotoxy(0,1);

lcd\_puts(lcd\_output);

meas\_state = dias\_Measure;

//then set timerun\_dias=0

//also reset count\_average for the next operation

count\_average=0;

timerun\_dias=0;

}

}

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

void diasmeasure(void)

{

ADMUX=0b00100000;//choose ADC1 for reading AC

//enable ADC and set prescaler to 1/128\*16MHz=125,000

//and set interupt enable

//and start a conversion

ADCSR = 0b11001111;

}//dias measure

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

void diascal(void)

{

ADMUX=0b00100001;//choose ADC1 for reading DC

//enable ADC and set prescaler to 1/128\*16MHz=125,000

//and set interupt enable

//and start a conversion

ADCSR = 0b11001111;

if(timecount>=200)

{

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("Dias\_Cal");

timecount=0;

}

}

void initialize(void)

{

//Initialize LCD

timecount=0;

lcd\_init(LCDwidth);

lcd\_clear();

lcd\_gotoxy(0,0);

lcd\_putsf("White: Start");

lcd\_gotoxy(0,1);

lcd\_putsf("Grey: Stop");

//set up timer0

TIMSK =2; //turn on timer 0 comp match

OCR0 = 250; //set the compare register to 250

//prescaler to 64 and turn on clear-on-match

TCCR0 = 0b00001011;

timepress0 = 0;

timepress1 = 0;

DDRB=0x00; //PORT B is an input(2 buttons)

DDRD=0xff; //PORT D is an output(motor control);

PORTD=0x00;

PORTB=0xff;

PORTA=0x00;

maxpressure = 160;

meas\_state = Sys\_Measure;

former=TH\_sys-0.01;

TH\_sys=4.0;

TH\_rate = 2.5;

TH\_dias = 4.8;

timerun\_dias=0;

time\_pulse=0;

timerate=0;

timing=40;

timedisplay=0;

total\_pulse\_period=0;

systolic=0;

diastolic=0;

pulse\_per\_min=0;

Vref=5.0;

sys\_count=0;

count\_average=0;

countpulse=0;

DC\_gain=213;

accum\_data=0;

press\_data=0;

count=0;

#asm

sei

#endasm

}