CPE301 – SPRING 2020

Design Assignment 5

Student Name: Jason Villanueva

Student #: 2001255420

Student Email: villaj21@unlv.nevada.edu

Primary Github address: <https://github.com/jasonvillanuevagit/submission_designAssignments->

Directory: <https://github.com/jasonvillanuevagit/submission_designAssignments-/tree/master/DesignAssignment5>

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

Atmel Studio 7.0 Atmega328PB-Xmini PC LM-35 Sensor

- Assembler -Multifunction Shield

- Simulator

- Debugger

A close up of text on a white background

Description automatically generated

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A**

#define *F\_CPU* 16000000UL

#include <avr/interrupt.h>

#include <avr/io.h>

#include <math.h>

#include <stdio.h>

#define BAUD 9600

#define BAUDPRESCALAR ((*F\_CPU*/(BAUD))/16 - 1)

#define bool unsigned int

#define true 0x01

#define false 0x00

#define rxEN 1

#define txIEN 1

#define drIEN 1

#define txEN 1

void sStringUART();

void CTCd1(*uint16\_t* delay, *uint8\_t* clk);

void CTCinit1();

void cfts(float floatNum);

float cftf(*int16\_t* fixed);

void OWinit();

void OWinput();

void OWoutput();

*uint8\_t* OWread();

void OWreset();

void OWw0();

void OWw1();

void DSc(*uint8\_t* command);

void convertT();

*int16\_t* readT();

void initUART(*uint8\_t* umsel, *uint8\_t* upm, *uint8\_t* usbs, *uint8\_t* ucsz);

void OWd1us();

float cctf(float celsius);

static char charC[20];

static volatile char\* ds;

static volatile bool uartBUS;

static volatile bool ctcBUS;

#define \_NOP() do { \_\_asm\_\_ \_\_volatile\_\_ ("nop"); } while (0)

#define OWP PORTC

#define OWddr DDRC

#define OWpin PINC

#define OWsh 4U

int main(void){

volatile *int16\_t* temperature = 0;

initUART(0x00, 0x00, 0x00, 0x03);

CTCinit1();

sei();

while (1){

convertT(); //convert temp

temperature = readT(); //read data string

ds = "Celsius: ";

sStringUART();

cfts(cftf(temperature)); //send C

sStringUART();

ds = "Fahrenheit: ";

sStringUART();

cfts(cctf(cftf(temperature))); //send F

sStringUART();

CTCd1(3124U, 0x4); //500ms delay

}

}

void OWw0(){

OWoutput();

OWP &= ~(1U<<OWsh);

CTCd1(110U, 0x2);

OWP |= (1U<<OWsh);

}

void OWw1(){

OWoutput();

OWP &= ~(1U<<OWsh);

OWd1us();

OWP |= (1U<<OWsh);

CTCd1(110U, 0x2);

}

//flt to str

void cfts(float floatNum){

*snprintf*(charC, sizeof(charC), "%f\r\n", floatNum);

ds = charC; //put new char into data string

}

void OWoutput(){

//output port

OWddr |= (1U<<OWsh);

}

void OWinput(){

//input port

OWP |= (1U<<OWsh);

OWddr &= ~(1U<<OWsh);

}

*uint8\_t* OWread(){

volatile *uint8\_t* input;

OWoutput();

OWP &= ~(1U<<OWsh);

OWd1us();

OWP |= (1U<<OWsh);

OWinput();

CTCd1(6U,0x2);

input = OWpin;

CTCd1(110U,0x2);

return ((input & (1U<<OWsh)) >> OWsh);

}

void OWreset(){

OWoutput();

OWP &= ~(1U<<OWsh);

CTCd1(949U, 0x2);

OWP |= (1U<<OWsh);

}

void OWinit(){

OWreset();

OWinput();

while((OWpin & (1U<<OWsh))){};

while(!(OWpin & (1U<<OWsh))){};

}

void OWd1us(){

\_NOP();

\_NOP();

\_NOP();

\_NOP();

\_NOP();

\_NOP();

\_NOP();

\_NOP();

}

void DSc(*uint8\_t* command){

volatile const *uint8\_t* maskArr[] = {0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80};

for(volatile *uint8\_t* i = 0; i < 8; i++){

if(command & maskArr[i]) OWw1();

else OWw0();

}

}

void convertT(){

OWinit();

DSc(0xCC);

DSc(0x44);

OWoutput();

OWP &= ~(1U<<OWsh);

OWd1us();

OWP |= (1U<<OWsh);

OWinput();

while(!(OWpin & OWsh));

}

*int16\_t* readT(){

volatile *int16\_t* temperature = 0;

OWinit();

DSc(0xCC);

DSc(0xBE);

for(volatile *uint8\_t* i = 0; i < 16; i++){

temperature |= (OWread()<<i);

}

OWreset();

return temperature;

}

float cftf(*int16\_t* fixed){

//fixed to float

volatile float tempFix = (float)((fixed & 0xFFF0) >> 4);

tempFix += ((fixed & 0x0001)>>0)\**pow*(2,-4) + ((fixed & 0x0002)>>1)\**pow*(2,-3) + ((fixed & 0x0004)>>2)\**pow*(2,-2) + ((fixed & 0x0008)>>3)\**pow*(2,-1);

return tempFix;

}

float cctf(float celsius){

//C to F

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

}

void sStringUART(){

//send string

uartBUS = true;

UCSR0B |= ((txIEN << TXCIE0) |

(drIEN << UDRIE0) |

(txEN << TXEN0)); //interrupts and tx

while(uartBUS){}; //wait

}

void CTCd1(*uint16\_t* delay, *uint8\_t* clk){

OCR1A = delay;

ctcBUS = true;

TCNT1 = 0x00;

TIMSK1 |= (1U<<1);

TCCR1B |= (clk);

while(ctcBUS);

}

void initUART(*uint8\_t* umsel, *uint8\_t* upm, *uint8\_t* usbs, *uint8\_t* ucsz){

UBRR0H = (*uint8\_t*)(0x00);

UBRR0L = (*uint8\_t*)(BAUDPRESCALAR); //baud rate

UCSR0C = (umsel << UMSEL00) |

(upm << UPM00) |

(usbs << USBS0) |

(ucsz << UCSZ00); //nitiate UCS

}

void CTCinit1(){

TCCR1B |= (1U<<3); //timer1 is CTC

}

ISR(USART0\_UDRE\_vect){

if(\*ds != '\0'){ //buffer ! empty

UDR0 = \*ds; //send data

ds++; //location++

}

else{

UCSR0B &= ~(drIEN << UDRIE0); //disable interrupt

uartBUS = false;

}

}

ISR(USART0\_TX\_vect){

//disable TX and interupt

UCSR0B &= ~((txIEN << TXCIE0) |

(txEN << TXEN0));

}

ISR(TIMER1\_COMPA\_vect){

//Timer1 delay ISR

ctcBUS = false;

TCCR1B &= ~(0x07);

TIMSK1 &= ~(1U<<1);

}

1. **DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A**

N/A

1. **SCHEMATICS**

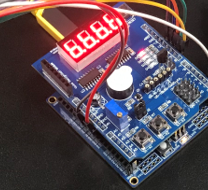
N/A

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

A screenshot of a computer

Description automatically generated

1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**



1. **VIDEO LINKS OF EACH DEMO**
2. **GITHUB LINK OF THIS DA**

<https://github.com/jasonvillanuevagit/submission_designAssignments-/tree/master/DesignAssignment5>

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

“This assignment submission is my own, original work”.

Jason Villanueva