CPE301 – FALL 2019

Design Assignment 3b

Student Name: Dillon Archibald
Student #: 5004439916
Student Email: Archid1@unlv.nevada.edu
Primary Github address: https://github.com/Dil-bert/Alabaster.git
Directory: DA3b/DA3B

Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

List of Components used

- 1x atmega 328p
- 3x wires
- 1x LM35 Temperature sensor
- 1x Breadboard
- 1x Prototype shield

Block diagram with pins used in the Atmega328P

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

```
#define F_CPU 16000000UL

#include <stdio.h>
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#define USART_BAUDRATE 9600
#define UBRR_VALUE (((F_CPU/(USART_BAUDRATE*16UL)))-1)
#define ADCINDEX 20
//store ADC values
uint8_t wave[ADCINDEX];
volatile uint8_t ii = 0;
volatile uint8_t flag = 0;

void InitPort();

void InitADC();
void SetADCChannel(uint8_t ADCchannel);
void StartADC(void);
void DisableADC(void);

void USART0Init(void);
void InitTimer0(void);
int USART0SendByte(char u8Data,FILE *stream);
void StartTimer0(void);
void StopTimer(void);

ISR(ADC_vect);

// Set Stream Pointer
FILE usart0_str = FDEV_SETUP_STREAM(USART0SendByte, NULL, _FDEV_SETUP_WRITE);

int main(void){
    // Initialize USART0
    USART0Init();
    // Initialize ports
    //TODO IMPLEMENT InitPort();
    InitPort();
    // Assign our stream to standard I/O streams
```

```
    stdout=&usart0_str;
    // Initialize ADC
    InitADC();
    // Select ADC channel
    SetADCChannel(5);
    // Initialize timer0
    InitTimer0();
    // Start timer 0
    StartTimer0();
    // Start conversion
    StartADC();
    // Enable global interrupts
    sei();
    while (1){
        //TODO add some sort of delay
        _delay_ms(1000);
        printf("ADC val[%u]=%u\r\n",ii,wave[ii]);
    }
}

void InitPort(){

}

void InitADC(){
    // Select Vref=Avcc and set left justified result
    ADMUX |=(1<<REFS0)|(1<<ADLAR);
    // Set prescaller to 32, enable auto triggering, enable ADC interrupt
    //  and enable ADC
    ADCSRA |=(1<<ADPS2)|(1<<ADPS0)|(1<<ADATE)|(1<<ADIE)|(1<<ADEN);
    // Set ADC trigger source - Timer0 compare match A
    ADCSRB |=(1<<ADTS1)|(1<<ADTS0);
}

void SetADCChannel(uint8_t ADCchannel){
    // Select ADC channel with safety mask
    ADMUX = (ADMUX & 0xF0)|(ADCchannel & 0x0F);
}

void StartADC(void){
    ADCSRA |=(1<<ADSC);
}

void DisableADC(void){
    ADCSRA &= ~((1<<ADEN)|(1<<ADIE));
}

void USART0Init(void){
    // Set baud rate
    UBRR0H = (uint8_t)(UBRR_VALUE>>8);
    UBRR0L = (uint8_t)UBRR_VALUE;
    // Set frame format to 8 data bits no parity, 1 stop bit
    UCSR0C |= (1<<UCSZ01)|(UCSZ00);
    // Enable transmission and reception
    UCSR0B |= (1<<RXEN0) | (1<<TXEN0);
}
```

```
void InitTimer0(void){
    // Set Initial Timer value
    TCNT0 = 0;
    // Place TOP timer value to Output compare register
    OCR0A = 99;
    // Set CTC mode
    //  and make toggle PD6/OC0A pin on compare match
    TCCR0A |= (1<<COM0A0)|(1<<WGM01);
}

int USART0SendByte(char u8Data,FILE *stream){
    // Wait while previous byte is completed
    while(!(UCSR0A & (1<<UDRE0))){};
    // Transmit Data
    UDR0 = u8Data;
    return 0;
}

void StartTimer0(void){
    // Set prescaller 8 and start timer
    TCCR0B |= (1<<CS01);
}

void StopTimer(void){
    TCCR0B &= ~(1<<CS01);
    TIMSK0 &= ~(1<<OCIE0A);
}

//ADC conversion complete ISR
ISR(ADC_vect){
    // clear timer compare match flag
    TIFR0 = (1<<OCF0A);
    // Toggle pin PD2 to track the end of ADC conversion
    PIND = (1<<PD2);
    wave[ii++] = ADCH;
    if(ii == ADCINDEX){ // read 20 values?
        StopTimer();
        DisableADC();
        flag = 1;
    }
}
```

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

MAIN:

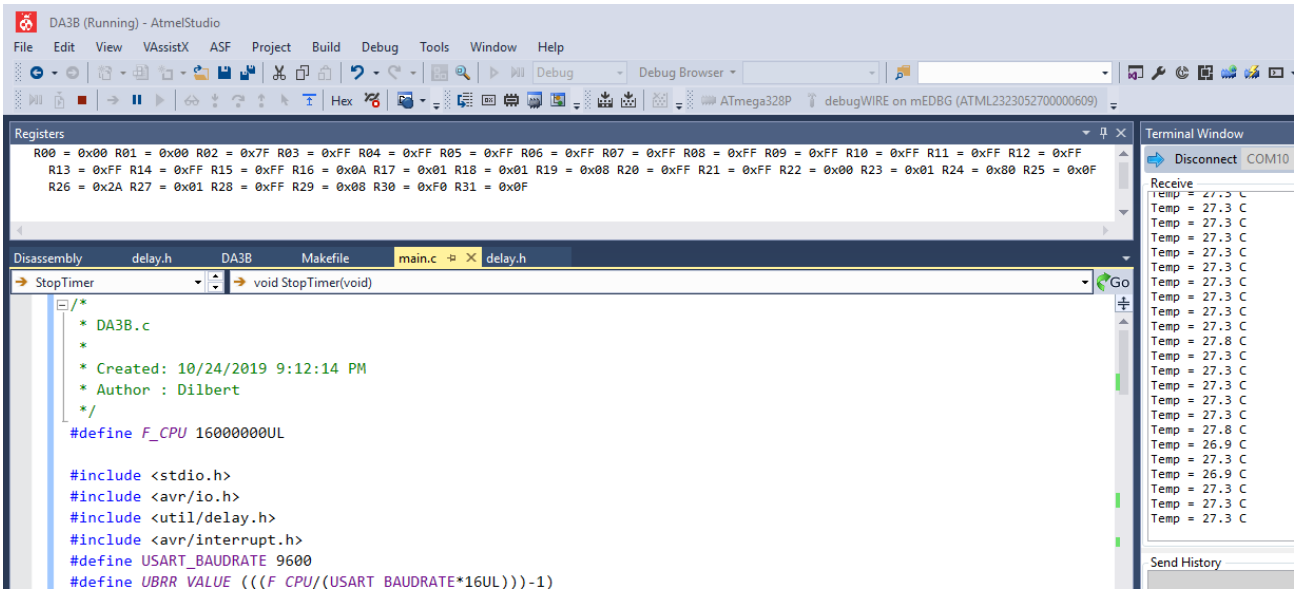
```
    while (1){
        _delay_ms(1000);
        tempOut = ((tempH << 8)|(tempL));
        tempOut = ((tempOut * 0.4883));
        printf("Temp = %.1f C\r\n",tempOut);
    }


void InitPort(){
    // Set pin C5 as an input pin
    DDRC |= (0<<PINC5);
    // Ensure pin C5 pull up resistor is off
    PORTC = (0<<PINC5);
    // Turn off Digital logic on pin C5
```

```
    DIDR0 |= (1<<ADC5D);
}

void InitADC(){
    // Select Vref=Avcc and set (left = ADLAR = 1)(right = ADLAR = 0) justified result
    ADMUX |=(1<<REFS0)|(0<<ADLAR);
    // Set prescaller to 32, enable auto triggering, enable ADC interrupt
    //  and enable ADC
    ADCSRA |=(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE)|(1<<ADIE)|(1<<ADEN);
    // Set ADC trigger source - Timer0 compare match A
    ADCSRB |=(1<<ADTS1)|(1<<ADTS0);
}

ISR(ADC_vect){
    // clear timer compare match flag
    TIFR0 = (1<<OCF0A);
    // Toggle pin PD2 to track the end of ADC conversion
    PIND = (1<<PD2);
    tempL = ADCL;
    tempH = ADCH;
    //if(ii == ADCINDEX){ // read 20 values?
    //    StopTimer();
    //    DisableADC();
    //    flag = 1;
    //}
}
```

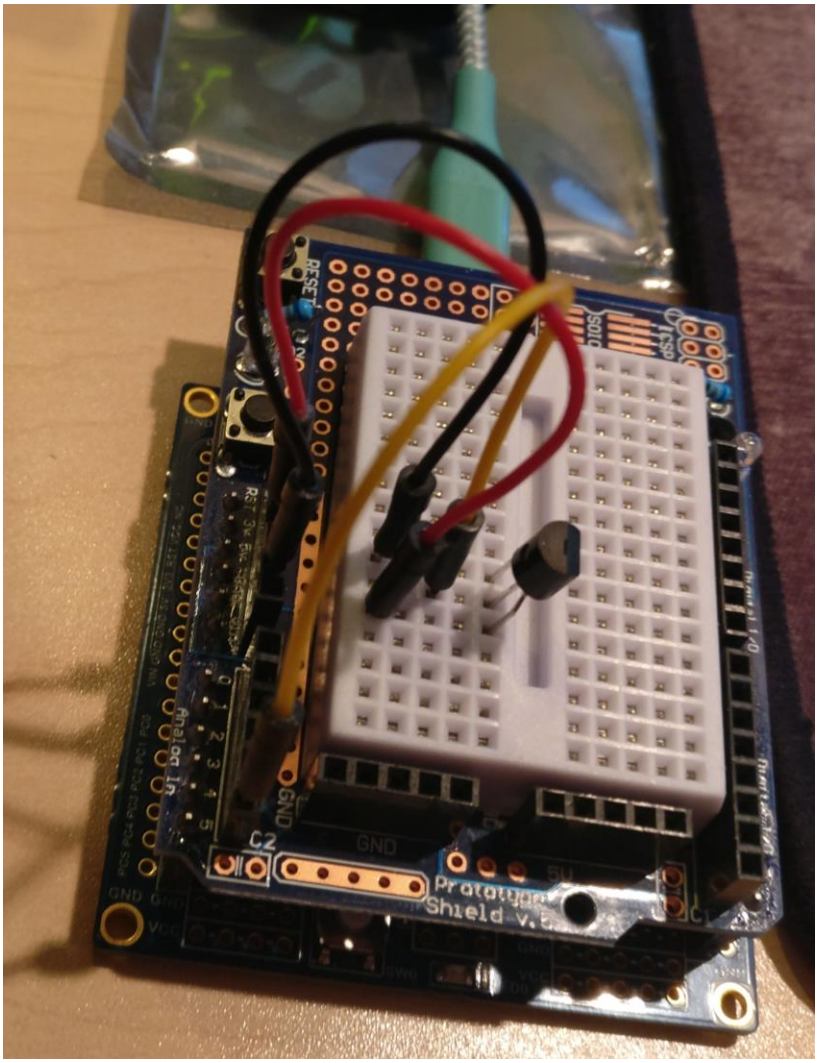
4. SCHEMATICS

Use fritzing.org

5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)



6. SCREENSHOT OF EACH DEMO (BOARD SETUP)



7. VIDEO LINKS OF EACH DEMO

https://youtu.be/d-bejH78b5A

8. GITHUB LINK OF THIS DA

https://github.com/Dil-bert/Alabaster/tree/master/DesignAssignments/DA3b/DA3B

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".
NAME OF THE STUDENT