CPE301 – SPRING 2019
Design Assignment 3B

Student Name: Yannick Kengne Tatcha
Student #: 5003294512
Student Email: kengneta@unlv.nevada.edu
Primary Github address: [Vasty1995/submission_da]
Directory: DA3B

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

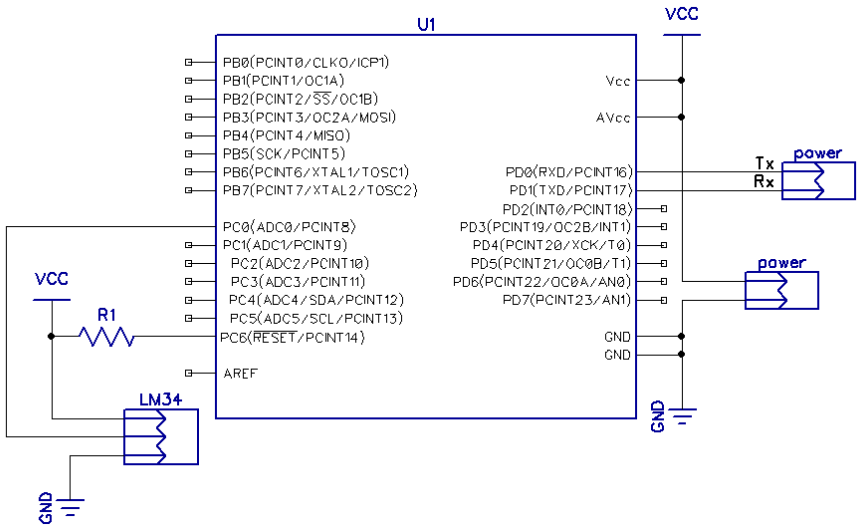
- Xplained mini board
- Micro usb
- ATMega329p

2. DEVELOPED CODE OF TASK 1/A in C

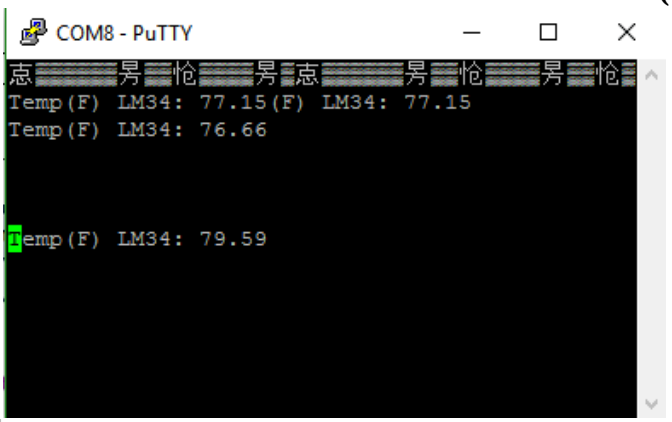
```
 * Author : YKengne
 */

#define     F_CPU       8000000UL
#include    <stdio.h>
#include    <avr/io.h>
#include    <util/delay.h>
#include    <avr/interrupt.h>

#define BAUDRATE    9600    // set baudrate
#define ASYNCH_NORM_PRESCALER (F_CPU/16/BAUDRATE - 1) // prescaler value

int USART0_sendChar(char, FILE*);   // Send character on USART0
void USART0_init (void);            // Initialize USART0
void ADC0_init();                   // to initiate ADC
void TIMER1_init();                 // initiate timer1

// reset stream pointer
// http://www.gnu.org/savannah-checkouts/non-gnu/avr-libc/user-manual/group__avr__stdio.html
FILE USART0_stream = FDEV_SETUP_STREAM(USART0_sendChar, NULL, _FDEV_SETUP_WRITE);

int main(void)
{

    stdout = &USART0_stream;   // change standard output to point to a USART stream

    USART0_init();      // Initiate USART0
    ADC0_init();        // Initiate ADC0
    TIMER1_init();      // Initiate TIMER1

    sei();              // enable global interrupts
    printf("\n");
    while(1);           // infinite loop. Wait for interrupt.

    return 0;
}

ISR(TIMER1_COMPA_vect)      // interrupt routine for Output Compare Match A.
// ISR that triggers at TIMER1_COMPT_vect
{
    uint16_t ADC_value = ADC;   // read ADC conversion
    float temperature_F;

    temperature_F = (ADC_value / 1024.0) * 5000;
    // 10 mV per degree Fahrenheit
    temperature_F = temperature_F / 10; // Convert to Fahrenheit
    printf("Temp(F) LM34: %3.2f\r", temperature_F);
}
```

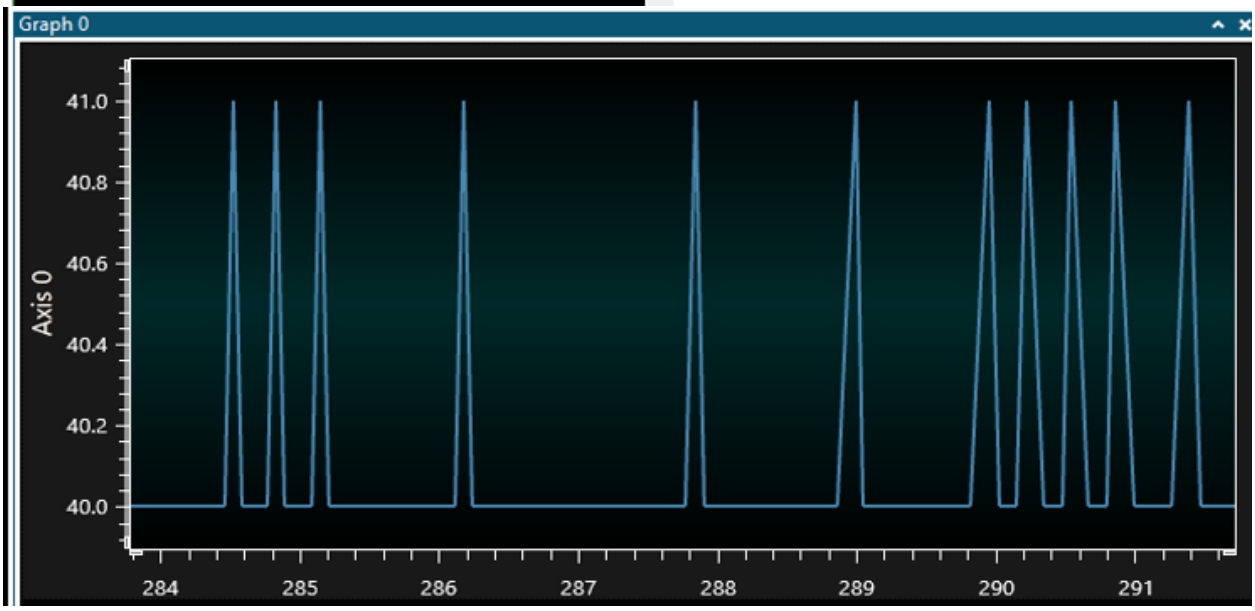
```
// TIMER1_init initiates TIMER1 for 1 second delays to trigger an interrupt at 1 Hz
{
    // WGM = 0100
    TCNT1  = 0;
    OCR1A  = 2*F_CPU/2/256-1;
    TIMSK1 = (1<<OCIE1A);
    TCCR1A = 0x00;
    TCCR1B = (1<<WGM12) | (1<<CS12) | (0<<CS11) | (0<<CS10);
}

void ADC0_init()
// ADC0_init will initiate ADC acquisition at ADC0
{
    DDRC &= ~(0<<DDC0);         // Clear bit 0 of DDRC
    ADMUX = 0; // use ADC0
    ADMUX |= (1 << REFS0);      // use AVcc as the reference
    ADCSRA |= (1 << ADPS2) | (1 << ADPS1)/* | (1 << ADPS0)*/; // 128 prescale for 16Mhz
    ADCSRA |= (1 << ADATE);     // Set ADC Auto Trigger Enable
    ADCSRB = 0;                 // 0 for free running mode
    ADCSRA |= (1 << ADEN);      // Enable the ADC
    ADCSRA |= (1 << ADSC);      // Start the ADC conversion
}

int USART0_sendChar(char data, FILE *stream)
/*
 * Procedure to send a single character over USART0. If character is linefeed, reset
 * line.
 * Assumes ASCII code.
 */
{
    if(data == '\n') // If linefeed, also print a return.
    {
        while(! (UCSR0A & (1<<UDRE0)) );// Wait for data register to be available.
        UDR0 = '\r';    // send return char to data register.
    }
    while(! (UCSR0A & (1<<UDRE0)) );    // Wait for data register to be available.
    UDR0 = data;        // send char to UART.
    return 0;
}

void USART0_init (void)
/*
 * Procedure to initialize USART0 asynchronous with enabled RX/TX, 8 bit data,
 * no parity, and 1 stop bit.
*/
{
    UCSR0B = (1<<TXEN0)  | (1<<RXEN0);  // enable transmit/receive
    UCSR0C = (1<<UCSZ01) | (1<<UCSZ00); // asynchronous, 8N1
    UBRR0L = ASYNCH_NORM_PRESCALER;     // To set 9600 baud rate with 8MHz clock
}
```

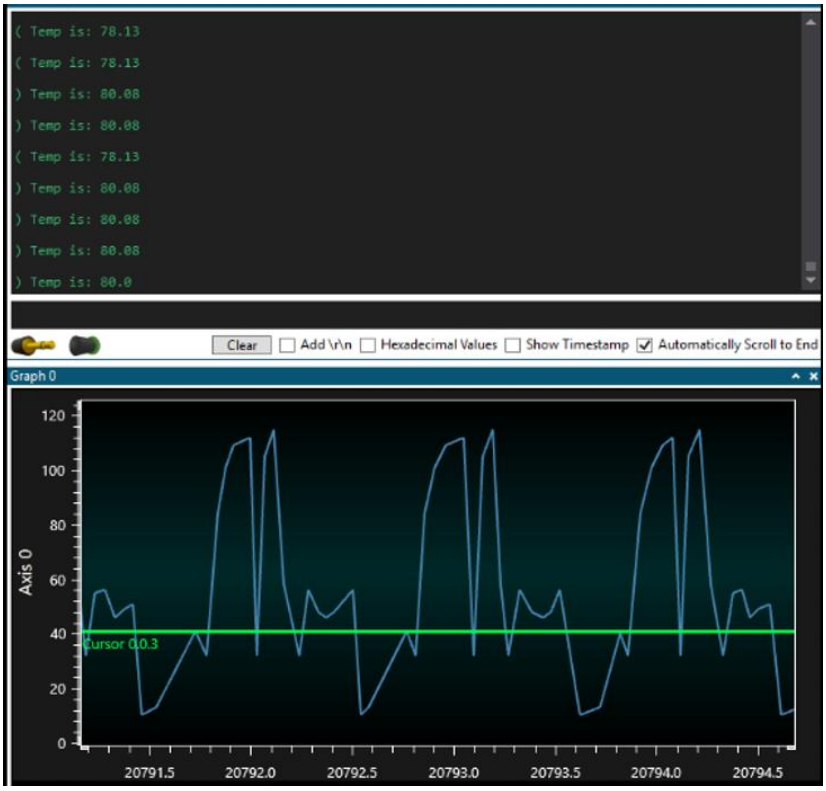
3. SCHEMATICS



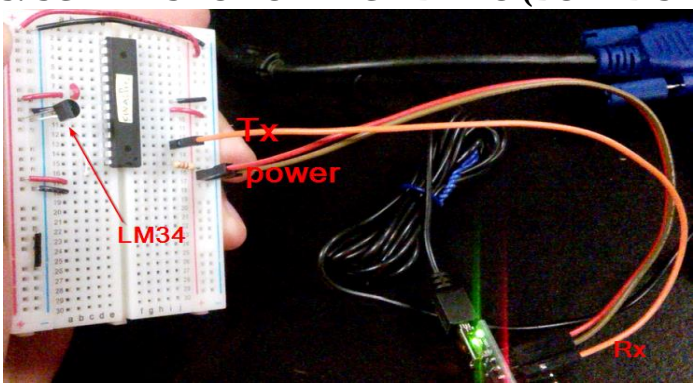
4. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)







5. SCREENSHOT OF EACH DEMO (BOARD SETUP)



6. VIDEO LINKS OF EACH DEMO

https://www.youtube.com/watch?v=YlmYH45lv3Y4

7. GITHUB LINK OF THIS DA

https://github.com/Vasty1995/submission_da/tree/master/DA3B

Student Academic Misconduct Policy

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

"*This assignment submission is my own, original work*". Yannick Kengne Tatcha