

CpE301 - Design Assignment 3

Design Assignment 3:

DUE: TBA

The goal of the assignment is to modify the above codes to do the following:

1. Write a C AVR program that will monitor the LM34/35 connected to an Analog pin to display the temperature in F on the serial terminal every 1 sec. Use a timer with interrupt for the 1 sec delay. Use a FTDI chip for serial to USB conversion.
2. Use the ATMEL Studio Data Visualizer or any Charting program to display the values in time.

Task 1: ADC 40%, Task 1: UART 40%, Task 2: 20%.

Submission:

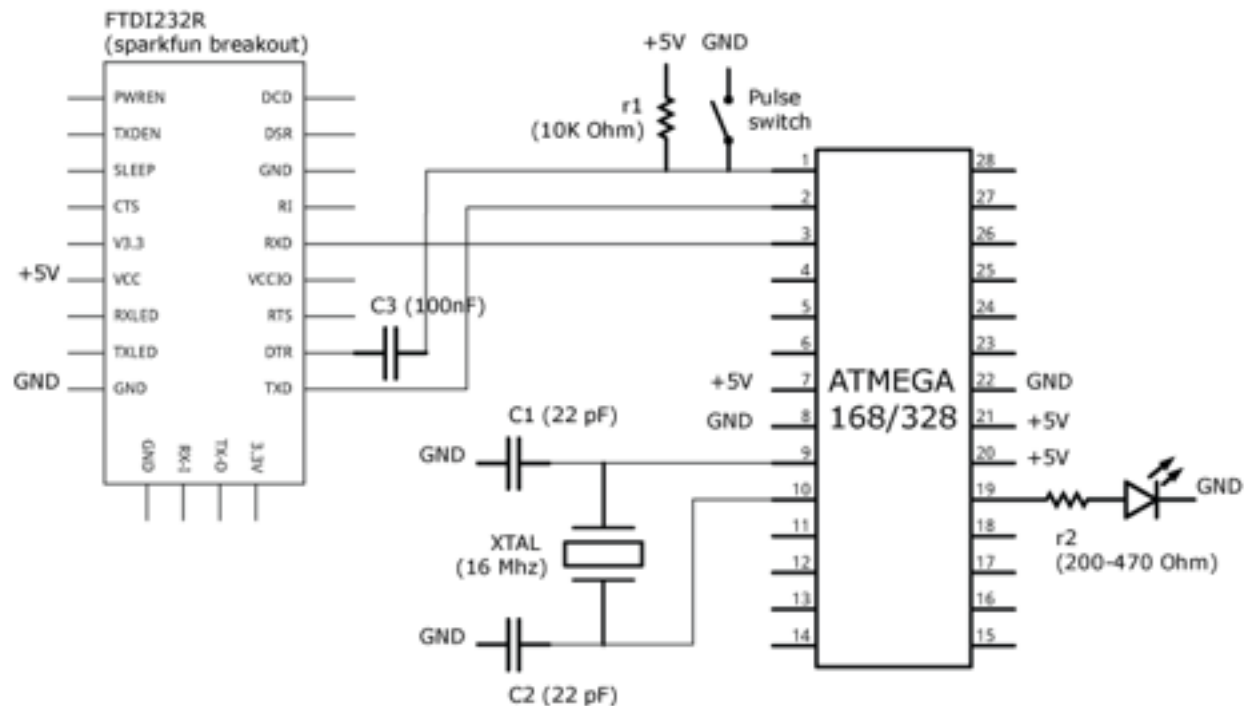
The following are required for successful completion of the design assignment:

- a. AVR C code that has been compiled and working.
- b. The C code should be well documented with explanation of every instruction.
- c. A word document that contains the flow chart of the assembly code along with the snapshots of the schematics, components connected on the breadboard and screen shoots.

NOTES:

Connecting the Microcontroller to PC using RS232-USB chip thro' USB connector.

Using a FTDI chip or Breakout Board - Use a 5V FTDI board.



Code Snippets:**USART CODE:**

// This code waits for a character and transmits the character back (with interrupts)

```
#include <avr/io.h>
#include <stdint.h>           // needed for uint8_t
#include <avr/interrupt.h>

#define FOSC 16000000          // Clock Speed
#define BAUD 9600
#define MYUBRR FOSC/16/BAUD -1

volatile char ReceivedChar;

int main( void )
{
    /*Set baud rate */
    UBRR0H = (MYUBRR >> 8);
    UBRR0L = MYUBRR;

    UCSRB |= (1 << RXEN0) | (1 << TXEN0);    // Enable receiver and transmitter
    UCSRB |= (1 << RXCIE0);                  // Enable receiver interrupt
    UCSRC |= (1 << UCSZ01) | (1 << UCSZ00);    // Set frame: 8data, 1 stp
    while(1)
    {
        ;                                // Main loop
    }
}

ISR (USART_RX_vect)
{
    ReceivedChar = UDR0;                  // Read data from the RX buffer
    UDR0 = ReceivedChar;                 // Write the data to the TX buffer
}

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

//set stream pointer
FILE usart0_str = FDEV_SETUP_STREAM(USART0SendByte, NULL, _FDEV_SETUP_WRITE);

printf("ADC ST_ARRAY[%u] = %u\r\n", index, array[index]);
-----//end FILE *stream -----
```

ADC CODE:

```
#include <avr/io.h>
#include <stdint.h>    // needed for uint8_t
#include <avr/interrupt.h>

volatile uint8_t ADCvalue; // Global variable, set to volatile if used with ISR
int main(void)
{
    ADMUX = 0;           // use ADC0
    ADMUX |= (1 << REFS0); // use AVcc as the reference
    ADMUX |= (1 << ADLAR); // Right adjust for 8 bit resolution
    ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); // 128 prescale for 16Mhz
    ADCSRA |= (1 << ADSC); // Set ADC Auto Trigger Enable
    ADCSRB = 0;           // 0 for free running mode
    ADCSRA |= (1 << ADEN); // Enable the ADC
    ADCSRA |= (1 << ADIFSC); // Enable Interrupts
    ADCSRA |= (1 << ADSC); // Start the ADC conversion
    sei(); // Thanks N, forgot this the first time =P
    while (1)
    {
    }
}

ISR(ADC_vect)
{
    ADCvalue = ADCH; // only need to read the high value for 8 bit
}
```