Eilat Avidan

CPE301 – SPRING 2018

Design Assignment 3

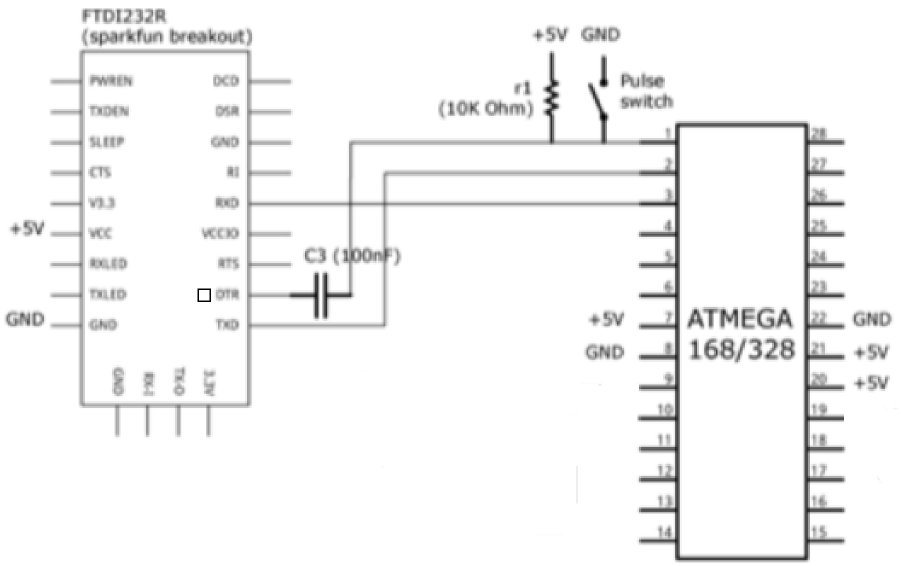
DO NOT REMOVE THIS PAGE DURING SUBMISSION:

The student understands that all required components should be submitted in complete for grading of this assignment.

NO	SUBMISSION ITEM	COMPLETED (Y/N)	MARKS (/MAX)
1	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		
2.	INITIAL CODE OF TASK 1/A		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 5/E		
4.	SCHEMATICS		
5.	SCREENSHOTS OF EACH TASK OUTPUT		
5.	SCREENSHOT OF EACH DEMO		
6.	VIDEO LINKS OF EACH DEMO		
7.	GOOGLECODE LINK OF THE DA		

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

- Atmega328P
- FTDI chip
- Mini USB to USB cable
- Switch
- Breadboard
- Resistors (mainly 100 ohms)
- LED (red)
- Power supply

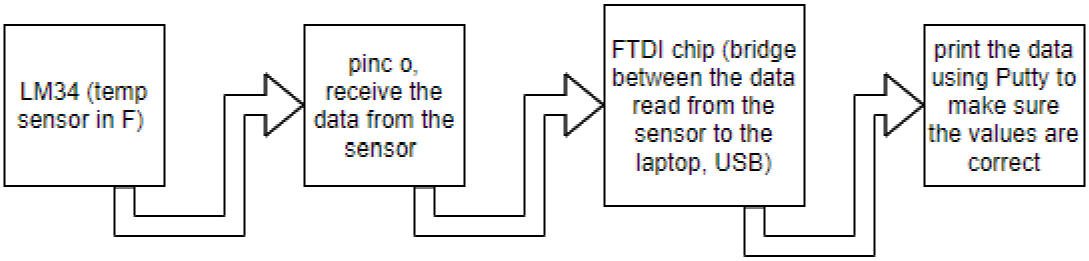


2. INITIAL/DEVELOPED CODE OF TASK 1/A

```
#define F_CPU 8000000UL
#define UBRR_9600 51 // for 8Mhz with .2% error


#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>
#include <avr/interrupt.h>
#include <stdint.h>

// Function Declarations
void read_adc(void);
void adc_init(void);
void USART_init( unsigned int ubrr );
void USART_tx_string( char *data );
volatile unsigned int adc_temp;
char outs[20];
```

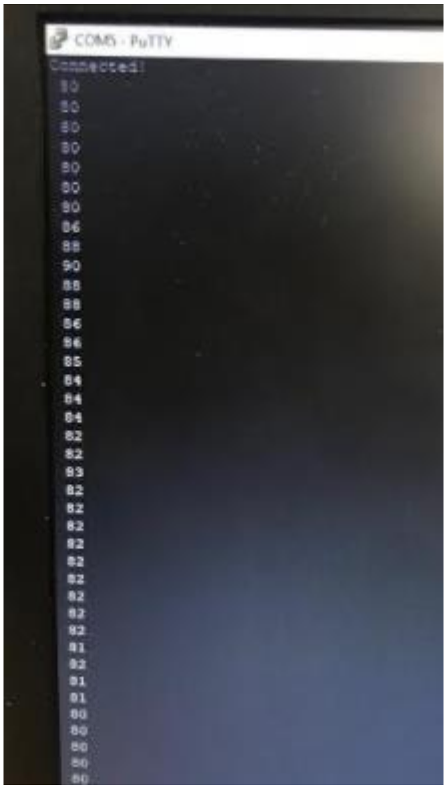
```
int main(void)
{
       adc_init();                 // Initialize the ADC (Analog / Digital Converter)
       USART_init(UBRR_9600);      // Initialize the USART (RS232 interface)
       USART_tx_string("Connected!\r\n");          // we're alive!
       _delay_ms(125);                                    // wait a bit
       TIMSK1 = (1<<TOIE1);                        //timer1 interrupt register
       TCNT1 = 34286;                                     //timer counts to this value
       TCCR1A = 0;
       TCCR1B = (1<<CS12);                                //256 prescaler
       sei();
       while(1)
       {
                            // wait for interrupt
       }
}

ISR (TIMER1_OVF_vect)                                     //overflow interrupt
{
              TCCR1B = 0;                                        //stop the counter
              read_adc();
              snprintf(outs,sizeof(outs),"%3d\r\n", adc_temp);// print ADC value
              USART_tx_string(outs);
              _delay_ms(125);                                    // wait a bit
              TCNT1 = 34286;
              TCCR1B = (1<<CS12);

}

void adc_init(void)
{
       /** Setup and enable ADC **/
       ADMUX = (0<<REFS1)|  // Reference Selection Bits
       (1<<REFS0)|                        // AVcc - external cap at AREF
       (1<<ADLAR)|                 // ADC left Adjust Result
       (0<<MUX2)|                         // Analog Channel Selection Bits
       (0<<MUX1)|                         // ADC0 Pin
       (0<<MUX0);

       ADCSRA = (1<<ADEN)|  // ADC ENable
       (1<<ADSC)|                         // ADC Start Conversion
       (1<<ADATE)|                 // ADC Auto Trigger Enable
       (0<<ADIF)|                         // ADC Interrupt Flag
       (0<<ADIE)|                         // ADC Interrupt Enable
       (1<<ADPS2)|                 // ADC Prescaler Select Bits
       (0<<ADPS1)|
       (1<<ADPS0);

}

/* READ ADC PINS*/
void read_adc(void)
{
       unsigned char i =4;                                //set i to 4- make 4 readings
       adc_temp = 0;                                      //initialize ADC_TEMP
       while (i--)
       {
              ADCSRA |= (1<<ADSC);
              while((ADCSRA & (1<<ADIF)) == 0);
              adc_temp += ADCH*2;                       //sum up 4 readings
              _delay_ms(50);
       }
       adc_temp = adc_temp / 4;                         // Average of four samples

}

/* INIT USART (RS-232) */


void USART_init( unsigned int ubrr )
{
       UBRR0H = (unsigned char)(ubrr>>8);
       UBRR0L = (unsigned char)ubrr;
       UCSR0B = (1 << TXEN0) | (1 <<RXEN0);             // Enable receiver, transmitter
       UCSR0C = (1 << UCSZ00) | (1 << UCSZ01);   //asynchronous 8-bit data 1 stop bit
}

/* SEND A STRING TO THE RS-232*/
void USART_tx_string( char *data )
{
       while ((*data != '\0'))                   //loop until the end of the string
       {
              while (!(UCSR0A & (1 <<UDRE0)));          //wait until flag set
              UDR0 = *data;
              _delay_ms(125);                                         // wait a bit
              data++;
       }
}
```

3. Flow chart



LM34 (temp sensor in F)
pinc o, receive the data from the sensor
FTDI chip (bridge between the data read from the sensor to the laptop, USB)
print the data using Putty to make sure the values are correct

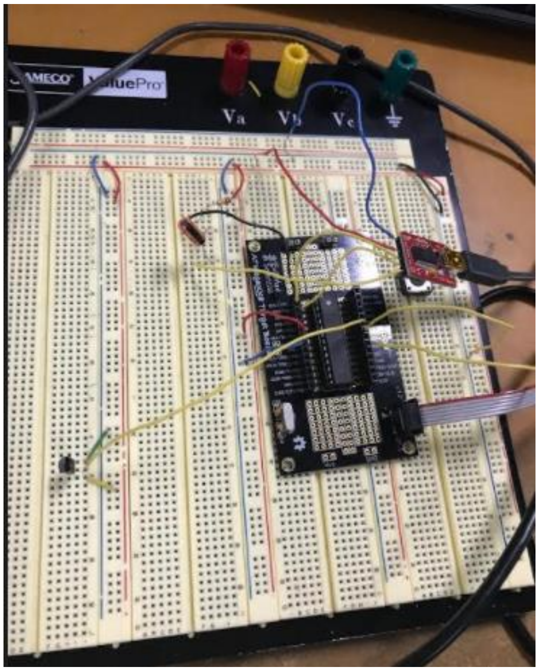
4. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)



```
Connected!
 80
 80
 80
 80
 80
 80
 80
 86
 88
 90
 88
 88
 86
 86
 85
 84
 84
 84
 82
 82
 83
 82
 82
 82
 82
 82
 82
 82
 82
 82
 81
 82
 81
 81
 80
 80
 80
 80
 80
```

Picture taking from Putty with the displayed values, in one point, we can see that the value increasing, I touched the sensor to make sure that the sensor is reading correctly.

5. SCREENSHOT OF EACH DEMO (BOARD SETUP)



6. VIDEO LINKS OF EACH DEMO

https://youtu.be/b5qD7vp0Rd0

7. GITHUB LINK OF THIS DA

git@github.com:EilatAvidan/microcon.git

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

"This assignment submission is my own, original work".
Eilat Avidan