Nathan Hanuscin

CPE301 – SPRING 2018

Design Assignment Midterm

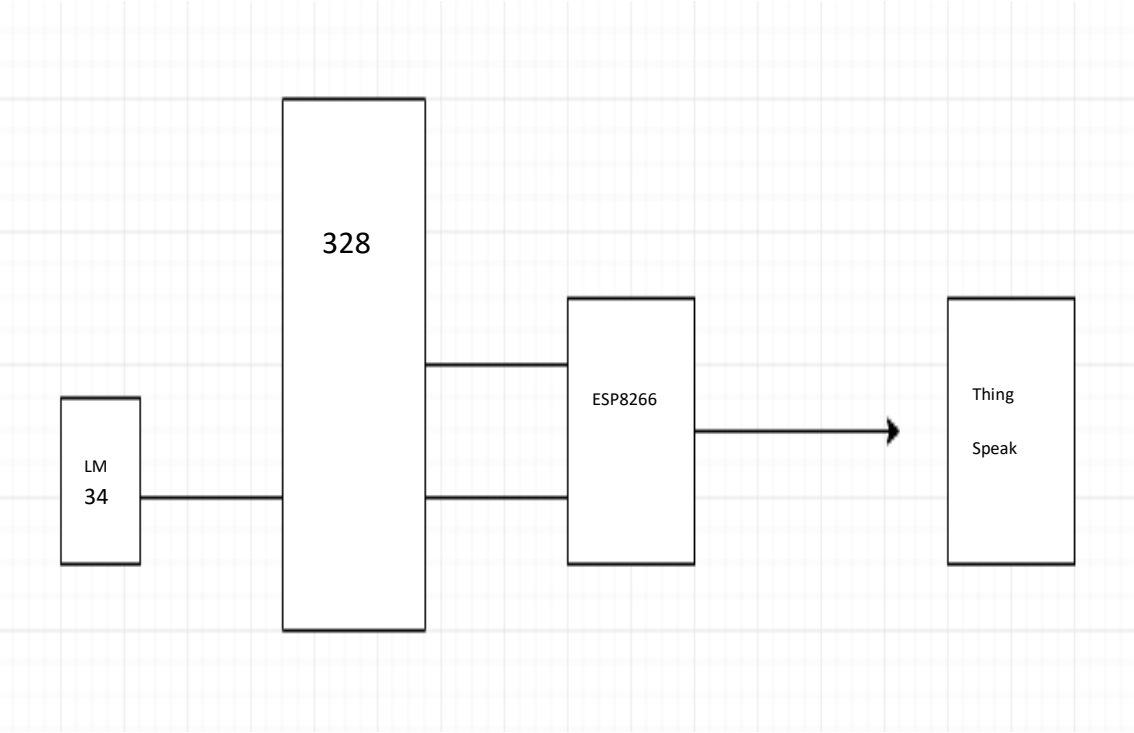
DO NOT REMOVE THIS PAGE DURING SUBMISSION:

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

LM34 Temp Sensor
ESP8266



2. INITIAL/DEVELOPED CODE OF TASK 1/A

Modified code from DA3

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

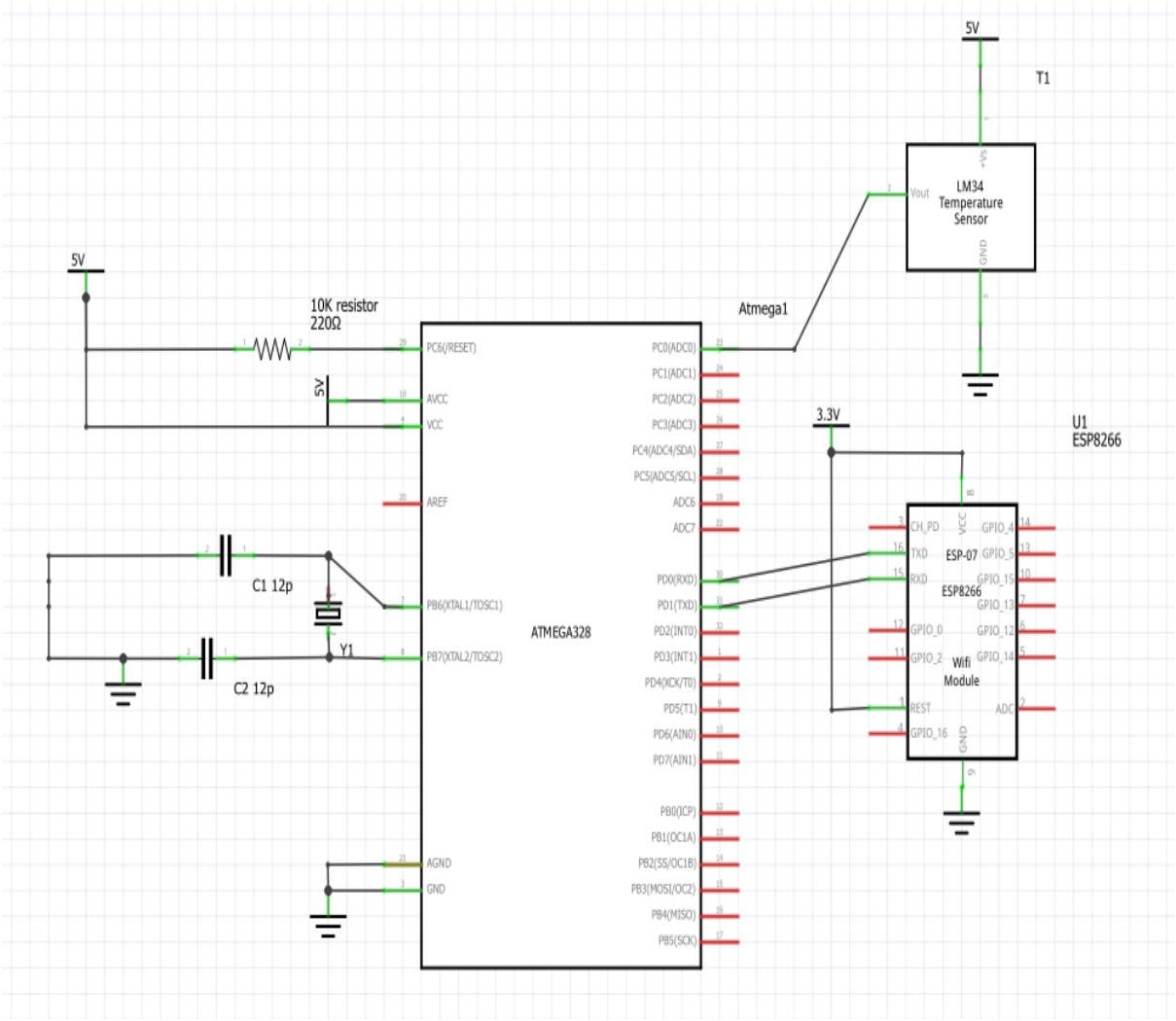
```
#define F_CPU 8000000UL
#define UBRR_115200 3 // for 8Mhz with 8.5% er

#define Domain "api.thingspeak.com"
#define API_Write_Key "PG5YKHOM60E8XQRI"
#define Channel_ID "461798"
#define SSID ""
#define Password ""


#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[30];
```

```
int main(void)
{
    adc_init();                         //Initialize the ADC (Analog / Digital Converter)
    USART_init(UBRR_115200);            //Initialize the USART (RS232 interface)
    _delay_ms(125);                     //wait a bit
    snprintf(outs,sizeof(outs),"AT\r\n");
    USART_tx_string(outs);
    _delay_ms(2000);
    snprintf(outs,sizeof(outs),"AT+CWMODE=3\r\n");  //wifi mode
    USART_tx_string(outs);
    _delay_ms(2000);
    snprintf(outs,sizeof(outs),"AT+CIPMUX=0\r\n");  //single connection
    USART_tx_string(outs);
    _delay_ms(2000);
    snprintf(outs,sizeof(outs),"AT+CIPMODE=0\r\n"); //normal mode
    USART_tx_string(outs);
    _delay_ms(2000);
    snprintf(outs,sizeof(outs),"AT+CWJAP=\"%s\",\"%s\"\r\n", SSID, Password);       //connect to wifi network
    USART_tx_string(outs);
    _delay_ms(2000);
    snprintf(outs,sizeof(outs),"AT+CPISTART=\"TCP\",\"%s\",80\r\n",Domain);         //connect to thingspeak
    USART_tx_string(outs);
    _delay_ms(2000);
```

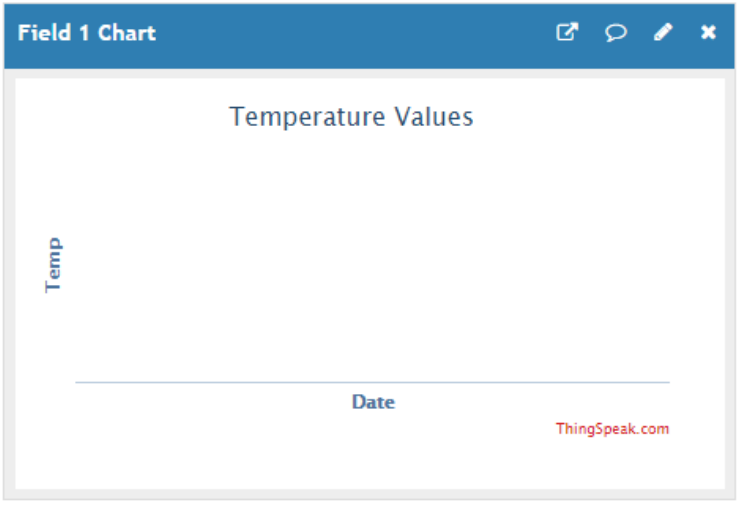
```
    while(1)
    {
        read_adc();
        snprintf(outs,sizeof(outs),"GET /update?api_key=%s&field1=1\r\n",API_Write_Key);        //send value
        USART_tx_string(outs);
        _delay_ms(150);
        snprintf(outs,sizeof(outs),"AT+CISEND=%3d\r\n",adc_temp);
        USART_tx_string(outs);
        _delay_ms(15000);           //wait 15 seconds
    }
}

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

    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 Pre-scaler of 64
    (1<<ADPS1)|
    (0<<ADPS0);

}
```

```
/* READ ADC PINS*/
void read_adc(void)
{
    unsigned char i = 4;                    //set for 4 ADC reads
    adc_temp = 0;                           //initialize temp to 0
    while (i--)
    {
        ADCSRA |= (1<<ADSC);                //start the conversion
        while((ADCSRA & (1<<ADIF)) == 0);   //wait for conversion to finish
        adc_temp += ADCH*2;                 //get temp value
        _delay_ms(50);                      //wait a bit
    }
    adc_temp = adc_temp / 4;                // Average a few samples

}

/* INIT USART (RS-232) */
void USART_init( unsigned int ubrr )
{
    UBRR0H = (unsigned char)(ubrr>>8);          //set baud rate
    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'))
    {
        while (!(UCSR0A & (1 <<UDRE0)));    //wait for the transmit buffer to empty
        UDR0 = *data;                       //put the data into the empty buffer, which sends the data
        _delay_ms(125);                     // wait a bit
        data++;
    }
}
```

4. SCHEMATICS

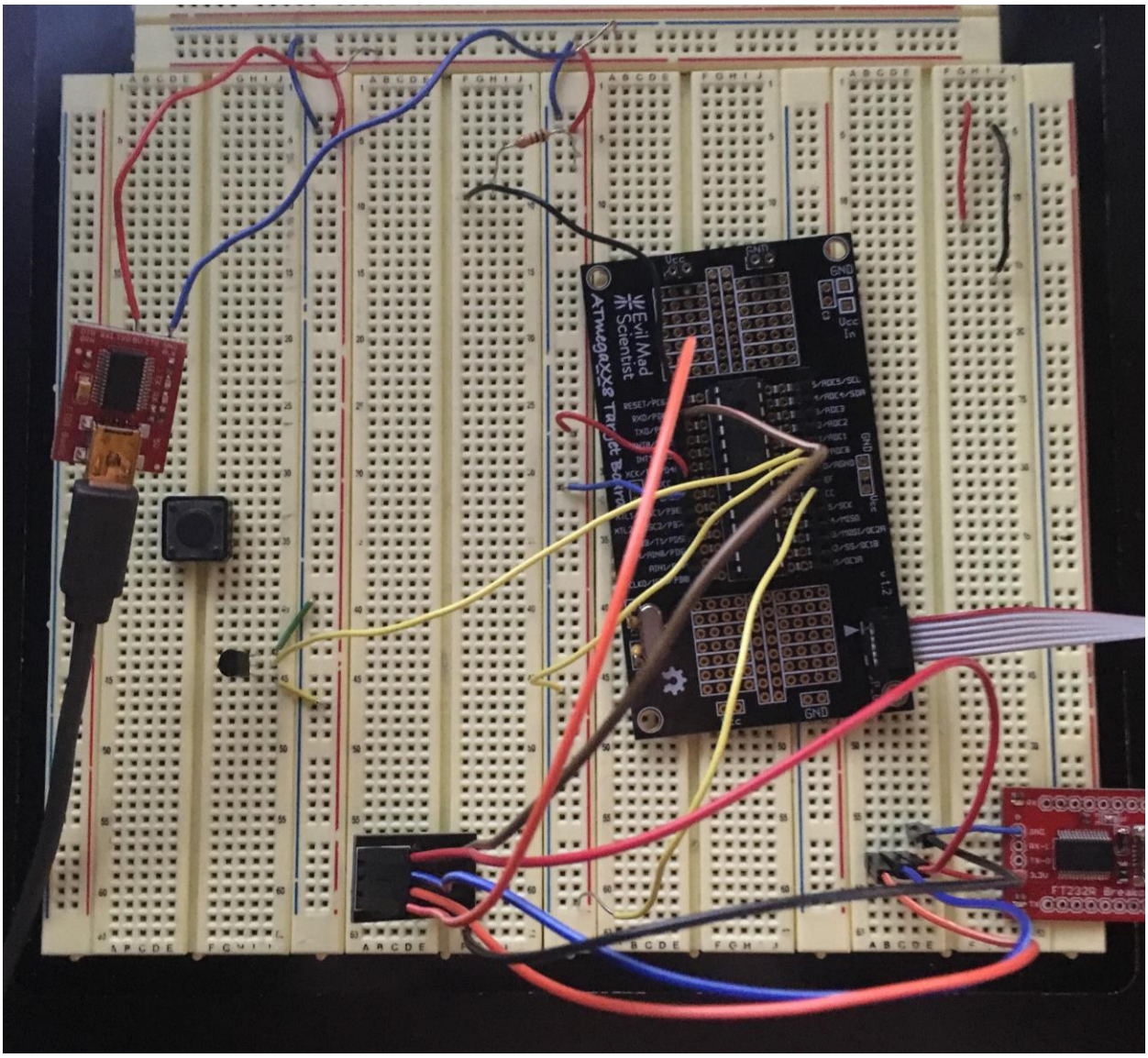


5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)



I couldn't figure out what was wrong with the ESP8266. I tried testing the AT commands in a putty terminal but it wasn't responding regardless of what baud rate I was using. I will try and solve this and hopefully complete this soon.

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)



7. VIDEO LINKS OF EACH DEMO

None yet.

8. GITHUB LINK OF THIS DA

https://github.com/nhanuscin/submit/tree/master/DA_Midterm

Student Academic Misconduct Policy

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

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

Nathan Hanuscin