CPE301 – FALL 2019

MIDTERM 1

Student Name: Jacob Patrick Reed
Student #: 1008448895
Student Email: reedj35@unlv.nevada.edu
Primary Github address: https://github.com/reedjacobp
Directory: https://github.com/reedjacobp/submission_da

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

- Atmega328PB Xplained Mini
- LM35
- ESP01
- Multifunction Shield

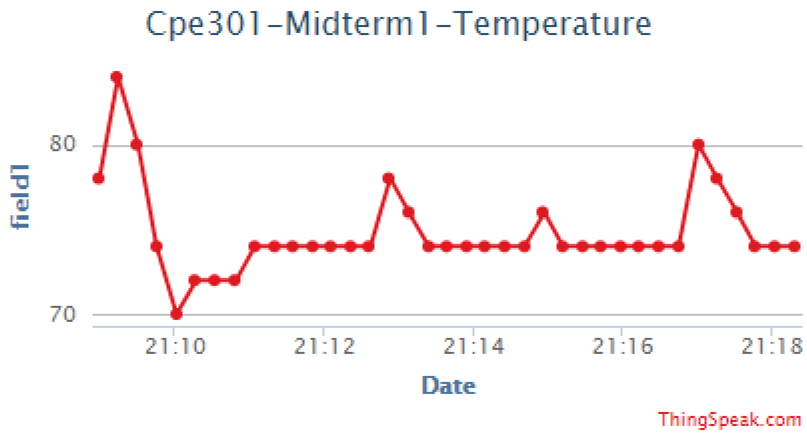
FTDI Vcc → ESP01 3.3V & EN
ATmega328PB Xplained Mini GND → ESP01 GND
Multifunction Shield APC220 Bluetooth Voice Recognition Module TX (Pin 4) → ESP01 RX
Multifunction Shield APC220 Bluetooth Voice Recognition Module TX (Pin 4) → ESP01 RX
Multifunction Shield PC4 GND (Pin 1) → LM35 GND (Pin 3)
Multifunction Shield PC4 Data (Pin 2) → LM35 Vout (Pin 2)
Multifunction Shield PC4 Vcc +5V (Pin 3) → LM35 Vs (Pin 1)

2. DEVELOPED CODE

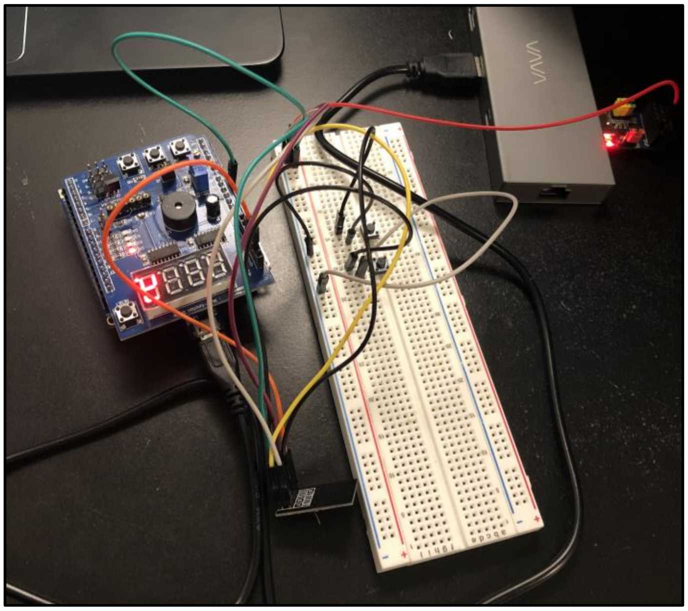
```
/*
 * Midterm1.c
 *
 * Created: 10/25/2019 2:07:45 PM
 * Author : jreed
 */

#define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <math.h>
#define BAUDRATE 115200
#define BAUD_PRESCALLER (int)round(((((double)F_CPU / ((double)BAUDRATE * 8.0))) - 1.0))

void USART_init(void); //function to initialize USART
void USART_send(unsigned char data); //function to send through USART
void USART_putstring(char* StringPtr); //goes through each character in a string to send
through USART
void ADC_init(void); //used to initialize analog to digital conversion

uint8_t OVF_COUNT = 0; //initialize the overflow count for interrupt
uint8_t OVF_LIMIT = 250; //set the limit the count can reach to set 1 sec delay

int main(void)
{
       USART_init(); //initialize USART
       ADC_init(); //initialize conversion
       USART_putstring("AT+CWMODE=1\r\n"); //sets ESP01 in station mode, all of the \r\n
seen throught the code is to "press enter/return" and create a new line
       _delay_ms(10); //the delays seen throughout the code is so that when strings are
sent, there's enough time to process
       USART_putstring("AT+CWJAP=\"SBG6700AC-8AD50\",\"98aa7a769e\"\r\n"); //set wifi and
password here
       _delay_ms(10);
       USART_putstring("AT+CIPMUX=0\r\n"); //sets ESP01 to have a single connection
       _delay_ms(10);

       TCCR0A = 0x00; //normal operation
       TCCR0B |= (1 << CS02); //set prescalar to 256
       TCNT0 = 16; //TOP = 256-250 = 16
       TIMSK0 |= (1 << TOIE0);
       sei(); //enable interrupt

       while (1)
       {
       }
}

void USART_init(void)
{
       UCSR0A = (1 << U2X0); //double USART transmission speed
       UBRR0H = (uint8_t)(BAUD_PRESCALLER >> 8); //set the baud rate register
       UBRR0L = (uint8_t)(BAUD_PRESCALLER);
       UCSR0B = (1 << RXEN0) | (1 << TXEN0); //enable transmit and receive
       UCSR0C = (3 << UCSZ00); //sets character size to 8-bits
}

void USART_putstring(char* StringPtr)
{
       while(*StringPtr != 0x00)
       {
              USART_send(*StringPtr);
              StringPtr++;
       }
}

void USART_send(unsigned char data) {                  // Function to transmit ASCII
value into UDR0
       while (!(UCSR0A & (1 << UDRE0)));               // Keep Checking until  UDRE0 data
register 'High' to break loop
       UDR0 = data;                                                  // Store unsigned
char serial data into UDR0
}
```

```
void ADC_init(void)
{
	ADMUX = (0<<REFS1)|  // reference selection bits
	(1<<REFS0)|   // AVcc - external cap at AREF
	(0<<ADLAR)|   // ADC Left Adjust Result
	(1<<MUX2)|    // Analog Channel Selection Bits
	(0<<MUX1)|    // ADC4 (PC4)
	(0<<MUX0);
	ADCSRA = (1<<ADEN)|  // ADC Enable
	(0<<ADSC)|    // ADC Start Conversion
	(0<<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);

}

ISR (TIMER0_OVF_vect)
{
	OVF_COUNT++; //increment the overflow counter
	if (OVF_COUNT == OVF_LIMIT) //check to see if the limit was reached
	{
		ADCSRA|=(1<<ADSC);   //start conversion
		while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish
		
		ADCSRA |= (1<<ADIF);
		
		int a = ADCL; //a is temperature
		a = a | (ADCH << 8);
		a = (a/1024.0) * 5000/10;
		a = (a*2)+32; //equation to convert celsius to fahrenheit. cannot use 9/5,
2 must be used
		a = a % 100;
		
		USART_putstring("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n");
//connect to thingspeak
		_delay_ms(10);
		USART_putstring("AT+CIPSEND=51\r\n"); //start sending data in transparent
transmission mode
		_delay_ms(10);
		USART_putstring("GET /update?key=YLVIWUJ8MIHKPKCR&field1="); //prepare to
send data using "Write" API Key
		_delay_ms(10);
		USART_send((a/10)+'0');
		_delay_ms(10);
		a = a % 10;
		USART_send((a)+'0');
		_delay_ms(10);
		USART_putstring(" ");
		_delay_ms(10);
		USART_putstring("\r\n");
		_delay_ms(10);
		OVF_COUNT = 0; //reset overflow counter
	}
	TCNT0 = 16; //reset TOP
}
```

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



4. SCREENSHOT OF EACH DEMO (BOARD SETUP)



5. VIDEO LINKS OF EACH DEMO

https://youtu.be/6JFHt2f-pCQ

6. GITHUB LINK OF THIS MIDTERM

https://github.com/reedjacobp/submission_da/tree/master/Midterms/Midterm%201

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

"This assignment submission is my own, original work".
Jacob Patrick Reed