Brian Lopez

CPE301 – SPRING 2018

Midterm 1

**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**

\*NOTE\* I was not able to flash the ESP8266 after hours of trying. I kept getting random characters. There were a few times where it did the boot message, but instead of ‘ready’, it said ‘invalid’. It let me type in commands, but there was no response when I entered it. Therefore I just did all I could.

* Atmega328P
* ESP8266 module
* Elego Voltage Regulator
* 9V battery
* LM34
* A few resistors and some jumper wires

1. **INITIAL/DEVELOPED CODE OF TASK 1/A**

#define BAUD 115200

#define *F\_CPU* 8000000UL

#define BAUDRATE ((*F\_CPU*)/(BAUD\*8UL)-1)

#include <avr/io.h>

#include <util/delay.h>

#include <stdio.h>

#include <avr/interrupt.h>

void init\_uart(){

// setting the baud rate based on FCPU and baudrate

UBRR0L = BAUDRATE;

// enabling TX & RX

UCSR0B = (1<<RXEN0)|(1<<TXEN0); // enable receive and transmit

UCSR0A = (1<<UDRE0)|(1<<U2X0);

UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // Set frame: 8data, 1 stop

}

void ADC\_init() {

ADMUX = 0; // read from port ADC0

ADMUX |= (1<<REFS0); // use AVcc for reference

ADCSRA |= (1<<ADPS2) | (1<<ADPS1); // prescalar of 64

ADCSRA |= (1<<ADEN); // enable ADC

ADCSRB = 0; // free running mode

}

unsigned int readADC()

{

ADMUX &= ~(1<<ADLAR); // clear the adc value

unsigned int val = 0;

ADCSRA |= (1 << ADSC); // start adc

while(ADCSRA & (1<<ADSC)); // wait until adc is done

val = ADC;

val = val \* 0.488; // doing (5 \* 100 \* adc) / 1024, just simplified

return val;

}

void usart\_send(unsigned char ascii)

{

while(!(UCSR0A & (1<<UDRE0))); // wait for transmit buffer to empty

UDR0 = ascii; // send the char

}

void AT\_send( unsigned char message[])

{

unsigned char i=0;

while(message[i] != '\0') // loop until NULL is reached

{

usart\_send(message[i]); // send the char

i++;

}

}

ISR (TIMER1\_COMPA\_vect)

{

int temp = 0;

temp = readADC(); // read from lm34

char out[30];

*snprintf*(out, 30, "GET /update?api\_key=DN72F6LKT2GVW8RD&field1=%d", temp); // get command set up to output temp

AT\_send(out); // send out the command

TIFR1 |= (1 << OCF1A); // reset interrupt flag

}

int main(void)

{

unsigned char AT[] = "AT\r\n";

unsigned char CIPMUX[] = "AT+CPIMUX=1\r\n";

unsigned char WIFI[] = "AT+CWJAP=\" SSID \", \"PASS\" \r\n";

unsigned char CIPSTART[] = "AT+CPISTART=0,\"TCP\",\"api.thingspeak.com\", 80\r\n";

*\_delay\_ms*(200);

init\_uart(); // initialize uart

*\_delay\_ms*(200);

ADC\_init(); // unitialize ADC

*\_delay\_ms*(200);

AT\_send(AT); // set up some commands

*\_delay\_ms*(2000);

AT\_send(CIPMUX);

*\_delay\_ms*(2000);

AT\_send(WIFI);

*\_delay\_ms*(2000);

AT\_send(CIPSTART);

*\_delay\_ms*(2000);

OCR1A = 3125; // set up the timer for 1 second CTC mode

TCCR1B |= (1 << WGM12);

TIMSK1 |= (1 << OCIE1A);

TCCR1B |= (1 << CS12);

sei(); // turn on interrupts

while(1)

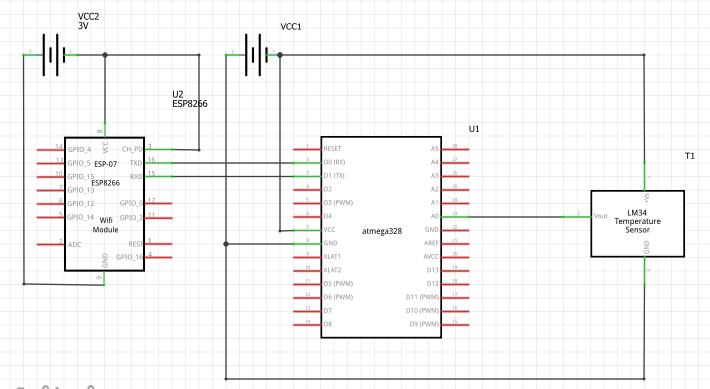
{

}

return 0;

}

1. **SCHEMATICS**



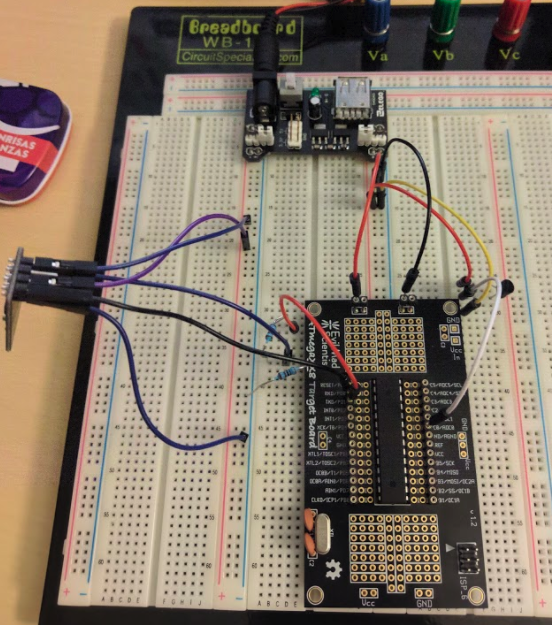
Using two different power supplies:

A 5V supply for the atmega and LM34 and a 3.3V for the ESP8266.

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

N/A, was not able to flash the ESP8266 properly ☹

1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**



Left power rails are 3.3V while the right power rails (right on top of the Atmega) are 5V

1. **VIDEO LINKS OF EACH DEMO**

N/A ☹

1. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

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

“This assignment submission is my own, original work”.

Brian Lopez