Your NAME

CPE301 – SPRING 2018

Design Assignment X

**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**

Components used:

* Atmega328P
* Lm34 temperature sensor
* Sparksfun FTDI Basic Breakout
* Lots of cable jumpers

Schematic is in Number 3

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

#define *F\_CPU* 8000000

#define BAUD 9600

#include <avr/io.h>

#include <util/delay.h>

#include <stdio.h>

#include <avr/interrupt.h>

volatile int count;

void init\_uart(){

// setting the baud rate based on FCPU and baudrate

UBRR0H =0x00;

UBRR0L =0x0C;

// 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

}

void USART\_Transmit( char \*data)

{

while((\*data != '\0')) { // transmits all chars but null

while(!(UCSR0A & (1<<UDRE0))); // waits for transmit flag to clear

UDR0 = \*data; // transmit next char

data++; // move to next char

}

}

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 \* 0.427; // doing (5 \* 100 \* adc) / 1024, just simplified

return val;

}

ISR (TIMER1\_COMPA\_vect)

{

int temp = 0;

char out[20]; // will have the string

count++; // increment counter

temp = readADC(); // get adc value

*snprintf*(out, sizeof(out), "Time(sec): %d\r\n", count);

USART\_Transmit(out);

*snprintf*(out, sizeof(out), "%3d F\r\n", temp); // put string into out

USART\_Transmit(out); // send string to be transmitted

TIFR1 |= (1<<OCF1A); // clear flag

}

int main (){

count = 0; // start count at 0

init\_uart(); // initialize UART

ADC\_init(); // initialize ADC

*\_delay\_ms*(250); // delay to wait for everything to catch up

OCR1A = 3125; // set compare value

TCCR1B |= (1 << WGM12); // set CTC mode

TIMSK1 |= (1 << OCIE1A); // enable timer interupt

TCCR1B |= (1<<CS12); // prescalar of 256

sei(); // enable interrupts

while(1)

{

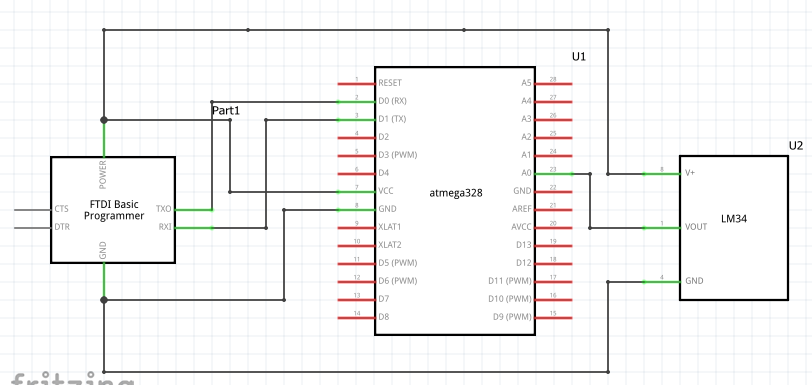
// do nothing

}

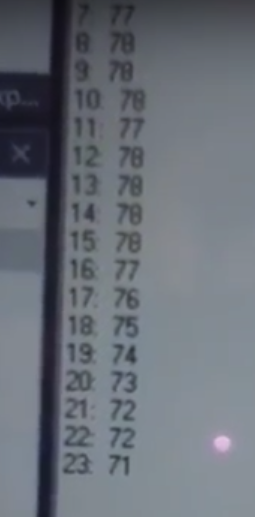
return 0;

}

1. **SCHEMATICS**

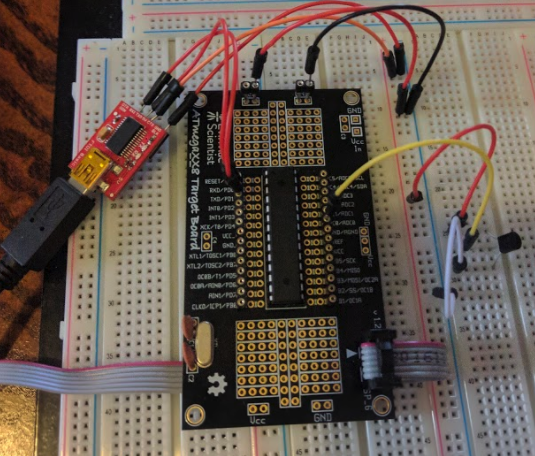


1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**



Format is <seconds> : Temperature. This screenshot shows it going down right after I placed a bag of ice water on top of the lm34 sensor.

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



1. **VIDEO LINKS OF EACH DEMO**

Video showing the circuit working:

<https://www.youtube.com/watch?v=uOaTSgg3JgQ>

**Student Academic Misconduct Policy**

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

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

Brian Lopez