Desi Battle

CPE301 – SPRING 2016

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.

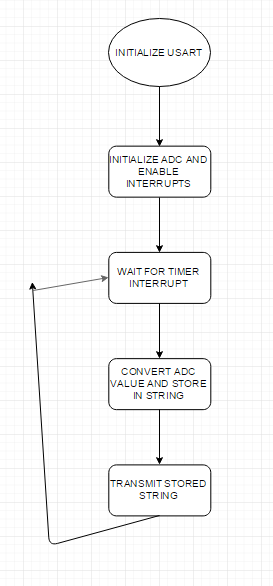
|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **SUBMISSION ITEM** | **COMPLETED (Y/N)** | **MARKS**  **(/MAX)** |
| 0. | COMPONENTS LIST & FLOW CHART |  |  |
| 1. | INITIAL CODE OF TASK 1/A |  |  |
| 2. | SCREENSHOT OF COMPONENTS |  |  |
| 3. | SCHEMATICS |  |  |
| 4. | SCREENSHOTS OF TASK OUTPUT |  |  |
| 5. | VIDEO LINKS OF EACH DEMO |  |  |
| 6. | GOOGLECODE LINK OF THE DA |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 0. | COMPONENTS LIST AND FLOW CHART |  |  |

LM34 – Weather sensor

FTDI - Basic

ATMEGA328P – Microcontroller



|  |  |  |  |
| --- | --- | --- | --- |
| 1. | INITIAL CODE OF TASK 1/A |  |  |

/\*

\* DA3.c

\*

\* Created: 3/31/2016 8:44:07 PM

\* Author: battled

\*/

#define F\_CPU 8000000UL //8 MHz xtal crystal

#include <avr/io.h> //

#include <avr/interrupt.h>

#include <string.h> //for string concatenation

#include <stdio.h> //for sprintf

//

void usart\_init (void)

{

UCSR0B = (1<<TXEN0); //transmit enable

UCSR0C = ((1<<UCSZ01)|(1<<UCSZ00)); //asynch mode

UBRR0L = 0x33; //baud rate 9600 at 8 MHz

}

void usart\_tx\_string(char \*data) //takes a string and sends it serially

{

while(\*data != '\0'){ //send chars until NULL is found

while(!(UCSR0A & (1<<UDRE0))); //wait for UDRE0 to be 1

UDR0 = \*data; //push current char into udr0 reg

data++; //point to next char

}

}

int main(void)

{

DDRC &= ~(1<<PORTC0); //PC0 is analog input

usart\_init(); //initialize USART

ADCSRA = ((1 << ADEN) | (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0));// ADC prescaler 128 ADEN

ADMUX = ((1<<REFS1)|(1 << REFS0)); // select internal 1.1 V Ref w/ ext cap at AREF pin and ADC0 (default)

sei(); //enable interrupts

//configure timer 1 to interrupt every second

TCNT1 = 65536 - ((double)F\_CPU/256); // set timer to overflow in 1 sec

TCCR1A = 0; //normal mode

TCCR1B = 4; //prescaler = 256

TIMSK1 |= (1<<TOIE1); //enable interrupt on overflow timer 1

while(1); //wait for interrupts

return 0;

}

ISR(TIMER1\_OVF\_vect) //timer1 overflow ISR

{

TCCR1B = 0; //stop timer 1

TIFR1 = 1; //clear overflow flag

int adc\_temp; //stores ADC temporarily

float adc\_tempf; //float for calculations

int adc\_tempi; //integer part

int adc\_tempd; //decimal part

//read ADC

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

while((ADCSRA &(1<<ADIF)) == 0); //wait for conversion to finish

adc\_temp = ADC; //save ADC value

adc\_tempf = (float)adc\_temp \* (1.1 / 1024) / 0.01; //(ADC\*res/.01) (lm34 sf = 10mv/degF)

adc\_tempi = (int)adc\_tempf; //integer part

adc\_tempf = adc\_tempf - adc\_tempi; //remove integer part of float temp

adc\_tempd = (int)(adc\_tempf \* 100); //store 2 decimal bits as float

//current temp: xxx.xx°F\r\n == 24 chars

char TmpTemp[24] = "Current Temp: ";

//sprintf here is used to "print" to TempInt string and then dot is added at end

sprintf(TmpTemp+14, "%d.%d°F\r\n", adc\_tempi,adc\_tempd);

//output degrees Fahrenheit

usart\_tx\_string(TmpTemp);

//reset time 1 for interrupt

TCNT1 = 65536 - ((double)F\_CPU/256); //overflow in 1 sec

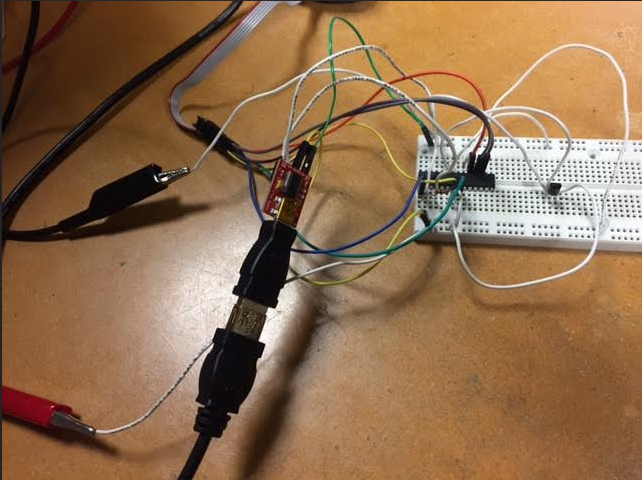
TCCR1A = 0; //normal mode

TCCR1B = 4; //prescaler = 256

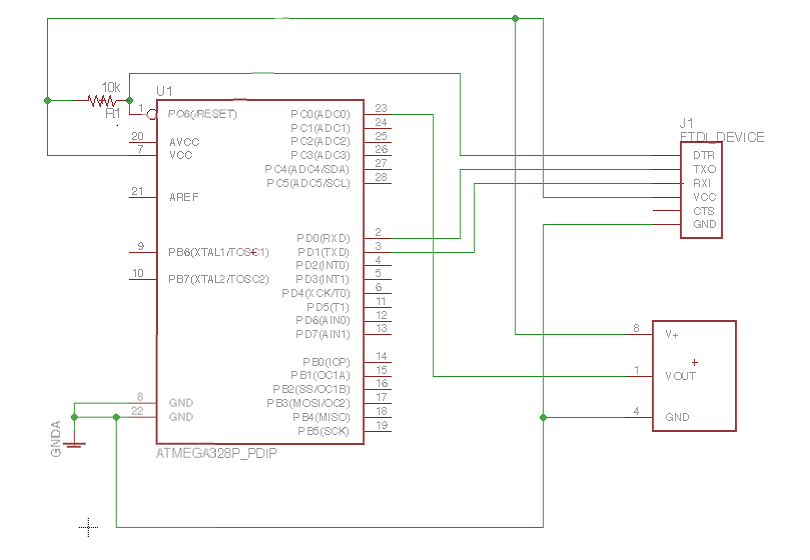
return;

}

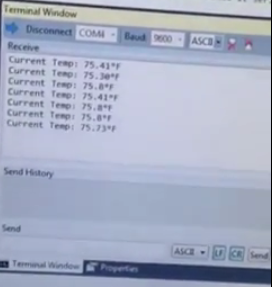
|  |  |  |  |
| --- | --- | --- | --- |
| 2. | SCREENSHOT OF COMPONENTS |  |  |



|  |  |  |  |
| --- | --- | --- | --- |
| 3. | SCHEMATICS |  |  |



|  |  |  |  |
| --- | --- | --- | --- |
| 4. | SCREENSHOT OF TASK OUTPUT |  |  |



|  |  |  |  |
| --- | --- | --- | --- |
| 5. | VIDEO LINKS OF EACH DEMO |  |  |
| https://www.youtube.com/watch?v=lyoo-tuWBYg | | | |
| 6. | GOOGLECODE LINK OF THE DA |  |  |
| <http://https://github.com/battled/DA0.git> | | | |

**Student Academic Misconduct Policy**

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

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

Desi Battle