CPE301 – SPRING 2019

Design Assignment 4A

Student Name: Cody Hudson

Student #: 8000603642

Student Email: hudsoc1@unlv.nevada.edu

Primary Github address: https://github.com/eed911/class\_proj.git

Directory:

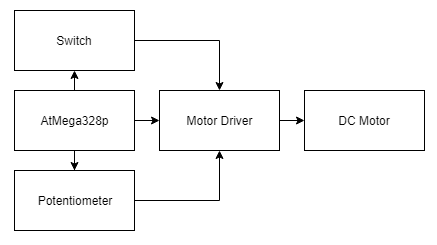
https://github.com/eed911/class\_proj/tree/master/DesignAssignments/DA4A/Project\_4A

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

List of Components used

* ATmega328P Xplained
* Switch
* Potentiometer
* Motor Driver
* DC Motor

Block diagram with pins used in the Atmega328P



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

1. Write a C AVR program that will monitor the LM34/35 connected to an Analog pin

(PC5) to display the temperature in F on the serial terminal every 1 sec. Use a timer

with interrupt for the 1 sec delay. Use a FTDI chip for serial to USB conversion.

2. Use the ATMEL Studio Data Visualizer or any Charting program to display the values

in time

Code:

/\*

\* DA4A.c

\*

\* Created: 4/11/2019 11:35:52 AM

\* Author : hudsoc1

\*/

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <stdio.h>

#include <util/delay.h>

#include <avr/interrupt.h>

// Define Baudrate for UART

#define BAUDRATE 9600

#define BAUD\_PRESCALLER (((*F\_CPU* / (BAUDRATE \* 16UL))) - 1)

// 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\_motor\_speed;

char outs[20];

int main(void)

{

adc\_init(); // Initialize the ADC

USART\_init(BAUD\_PRESCALLER); // Initialize the USART

USART\_tx\_string("Connected!\r\n"); // we're alive!

*\_delay\_ms*(125); // wait a bit

DDRD = 0xFF; //DDRD = (1<<3); // Set Port D as an output direction THIS IS REQUIRED TO OUTPUT THE PWM

// In the next line of code, we:

// 1. Set the compare output mode to clear OC2A and OC2B on compare match.

// To achieve this, we set bits COM2A1 and COM2B1 to high.

// 2. Set the waveform generation mode to fast PWM (mode 3 in datasheet).

// To achieve this, we set bits WGM21 and WGM20 to high.

TCCR2A = \_BV(COM2A1) | \_BV(COM2B1) | \_BV(WGM21) | \_BV(WGM20);

// In the next line of code, we:

// 1. Set the waveform generation mode to fast PWM mode 7 —reset counter on

// OCR2A value instead of the default 255. To achieve this, we set bit

// WGM22 to high.

// 2. Set the prescaler divisor to 1, so that our counter will be fed with

// the clock's full frequency (16MHz). To achieve this, we set CS20 to

// high (and keep CS21 and CS22 to low by not setting them).

TCCR2B = \_BV(WGM22) | \_BV(CS20) | \_BV(CS21) | \_BV(CS22);

// OCR2A holds the top value of our counter, so it acts as a divisor to the

// clock. When our counter reaches this, it resets.

OCR2A = 255;

// This is the duty cycle. Think of it as the last value of the counter our

// output will remain high for. Can't be greater than OCR2A of course.

OCR2B = 0;

PCMSK1 |= (1<<PCINT10);

PCICR |= (1<<PCIE1);

sei();

while (1)

{

read\_adc();

*\_delay\_ms*(250); // Why delay?

adc\_motor\_speed = adc\_motor\_speed/4;

*snprintf*(outs,sizeof(outs),"%3d\r\n", adc\_motor\_speed); // print it

USART\_tx\_string(outs); // Print random integer number

OCR2B = adc\_motor\_speed; // Replace with adc value

}

}

/\* INIT ADC \*/

void adc\_init(void)

{

/\*\* Setup and enable ADC \*\*/

ADMUX = (0<<REFS1)| // Reference Selection Bits

(1<<REFS0)| // AVcc - external cap at AREF

(0<<ADLAR)| // ADC Left Adjust Result

(0<<MUX2)| // Analog Channel Selection Bits

(0<<MUX1)| // ADC0 (PC0 PIN27)

(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);

}

/\* READ ADC PINS \*/

void read\_adc(void)

{

unsigned char i = 4;

adc\_motor\_speed = 0;

while (i--)

{

ADCSRA |= (1<<ADSC);

while(ADCSRA & (1<<ADSC));

adc\_motor\_speed+= ADC;

*\_delay\_ms*(50);

}

adc\_motor\_speed = adc\_motor\_speed / 4; // Average a few samples

}

/\* INIT USART (RS-232) \*/

void USART\_init( unsigned int ubrr )

{

UBRR0H = (unsigned char)(ubrr>>8);

UBRR0L = (unsigned char)ubrr;

UCSR0B = (1 << TXEN0); // Enable RX, TX & RX interrupt

UCSR0C = (3 << UCSZ00); //asynchronous 8 N 1

}

/\* SEND A STRING TO THE RS-232 \*/

void USART\_tx\_string( char \*data )

{

while ((\*data != '\0'))

{

while (!(UCSR0A & (1 <<UDRE0)));

UDR0 = \*data;

data++;

}

}

ISR(PCINT1\_vect){

DDRD ^= 0xFF;

*\_delay\_ms*(1000); //Ground Bounce

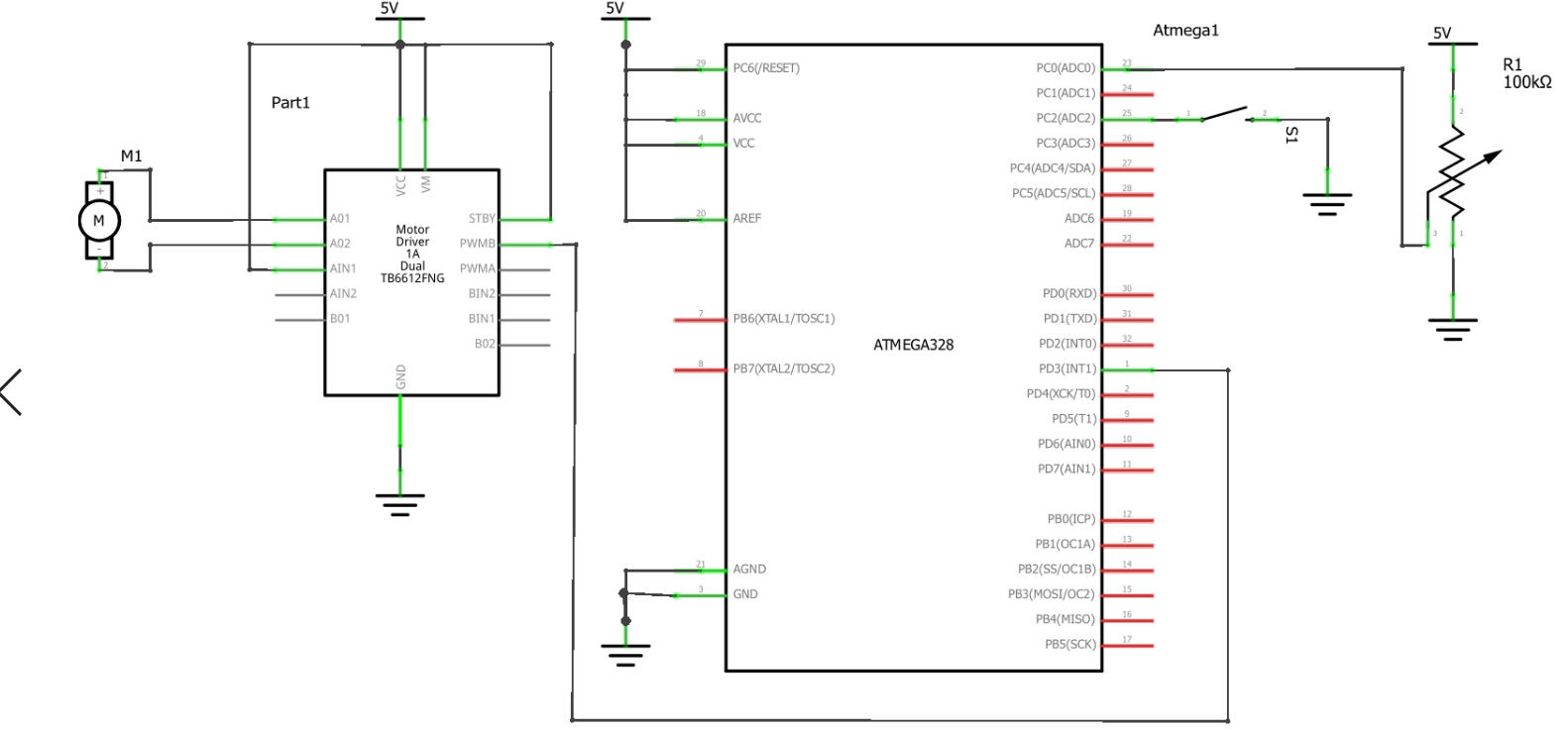
char check = "Pin Change ";

USART\_tx\_string(check); // Print random integer number

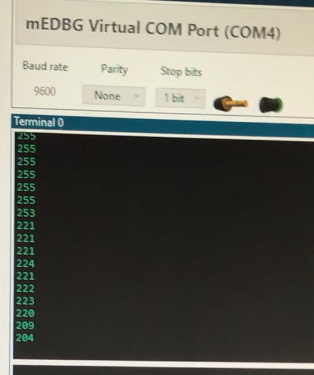
// Check if I bit is set in IREG

}

1. **SCHEMATICS**

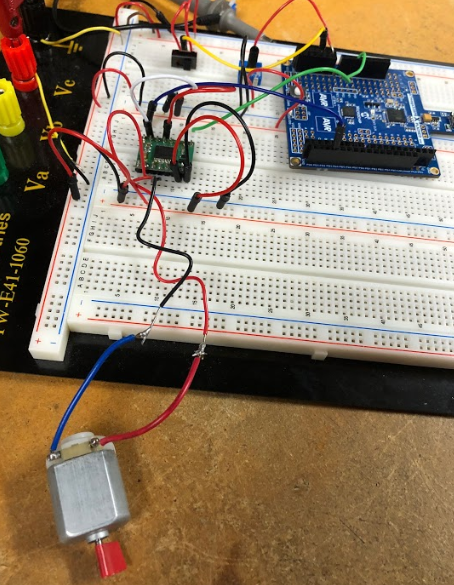
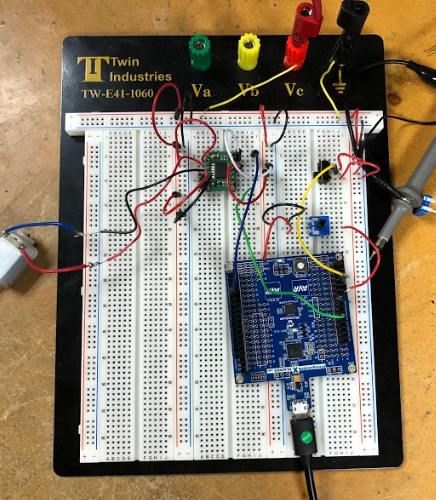


1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**



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

Screenshot of demo:

1. **VIDEO LINKS OF EACH DEMO**

Demo1:

<https://youtu.be/qg6q-xfl8Aw>

1. **GITHUB LINK OF THIS DA**

https://github.com/eed911/class\_proj.git

**Student Academic Misconduct Policy**

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

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

Cody Hudson