Brian Lopez

CPE301 – SPRING 2018

Design Assignment 4

**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**

* Atmega328P
* FTDI (only for providing power)
* DC motor, Stepper motor, and servo motor for tasks 1, 2, and 3 respectively
* Potentiometer
* Jumper wires

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

#define *F\_CPU* 8000000

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

int val = 0;

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

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

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

return ADC;

}

int main(void)

{

ADC\_init();

DDRB |= (1 << 1) | (1 << 2); // set pins portb.1 and 2 as output

PORTB = (1 << 2); // set B2 as high for clockwise rotation

DDRD |= (1 << 6); // set D6 as output to control enable on driver

OCR0A = 243;

TCCR0A |= (1 << COM0A1); // non-inverted mode

TCCR0A |= (1 << WGM02) | (1 << WGM00); // phase correct PWM

TCCR0B |= (1 << CS01); // prescalar of 8

while(1)

{

val = readADC() / 4.20; // read ADC and divide by 4.2 to not exceed 243

OCR0A = val; // which is 95% of PWM value. then update PWM compare

}

}

1. **TASK 2 CODE**

#define *F\_CPU* 8000000

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

volatile int step;

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

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

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

return ADC;

}

ISR(TIMER0\_COMPA\_vect)

{

if (step == 0) // change position of motor depending

PORTB = (1 << 1); // on which step it is currently at

else if (step == 1)

PORTB = (1 << 2);

else if (step == 2)

PORTB = (1 << 3);

else if (step == 3)

PORTB = (1 << 4);

step++;

if (step == 4) // increment and reset step if necessary

step = 0;

*\_delay\_us*(500); // small delay to let motor move

OCR0A = 255 - readADC() / 4.02; // change Compare value to ADC value

// divide 4.02 so that max value is 254.

TIFR0 |= (1 << OCF0A); // reset interrupt flag

}

int main(void)

{

ADC\_init(); // initialize ADC

DDRB = 0xFF; // set all port b ports as output

//int val = 0;

step = 0;

TCCR0A |= (1 << WGM01); // CTC mode

TCCR0B |= (1 << CS01) | (1 << CS00); // Prescalar of 64

TIMSK0 |= (1 << OCIE0A); // turn on overflow interrupt

OCR0A = 200; // set initial match value

sei(); // turn on interrupt

while (1)

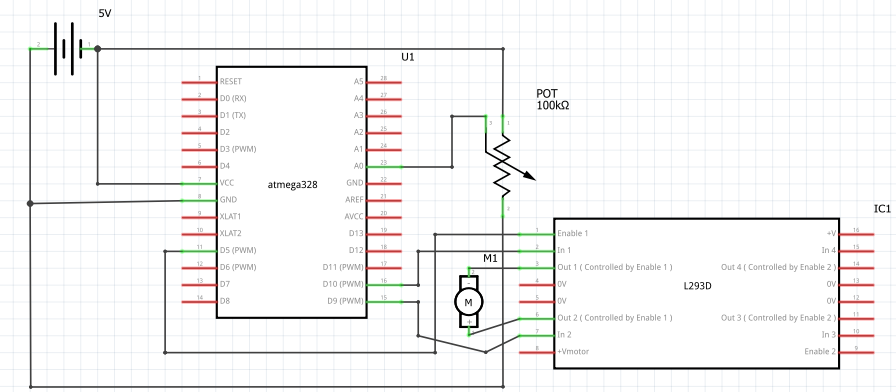
{

}

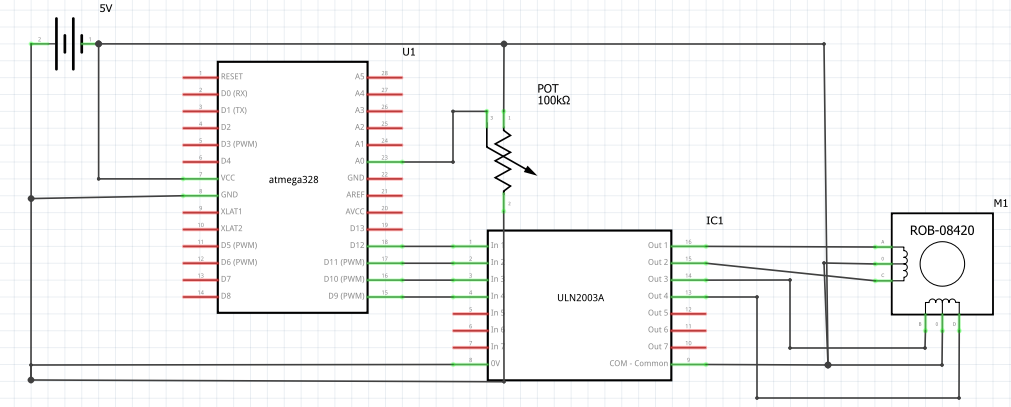
}

1. **SCHEMATICS**

**Task1:**



**Task2:**

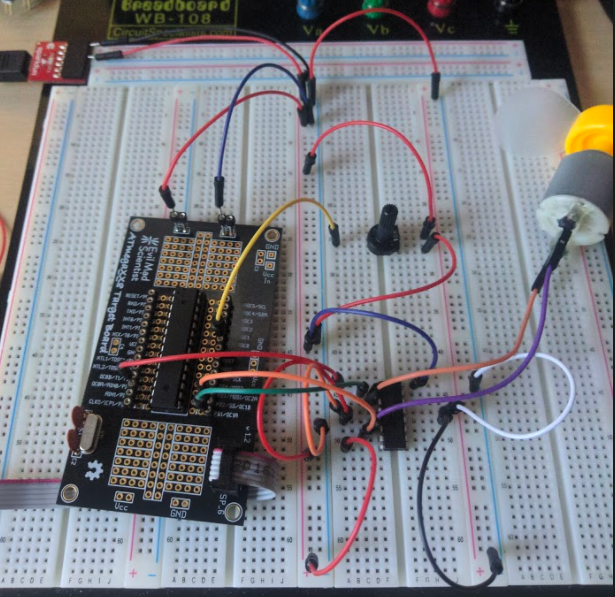


1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

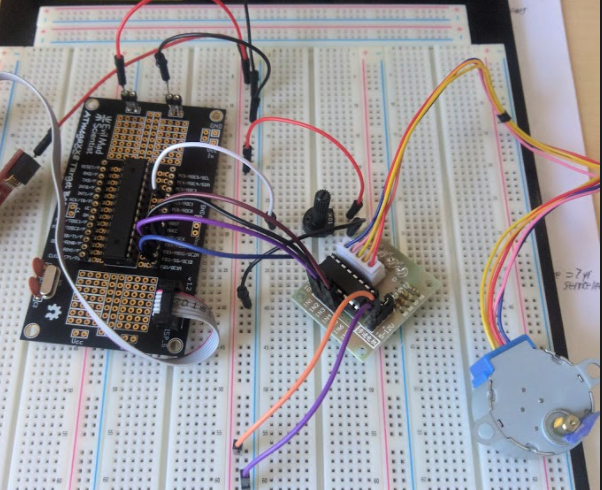
N/A

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

**Task1:**



**Task2:**

****

1. **VIDEO LINKS OF EACH DEMO**

Task 1: <https://www.youtube.com/watch?v=_ncwYZNwnsk>

Task 2: <https://www.youtube.com/watch?v=FJHlcehPWT0>

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