Audie Escala

CPE301 – SPRING 2018

Design Assignment 04

**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. | INITIAL CODE OF TASK 2/B |  |  |
| 4. | INITIAL CODE OF TASK 3/C |  |  |
| 5. | FLOW CHART OF EACH TASK |  |  |
| 6. | SCEENSHOTS OF EACH TASK SCHEMATIC |  |  |
| 5. | SCREEN SHOTS OF EACH TASK ON BREADBOARD |  |  |
| 7. | YOUTUBE/GOOGLECODE LINK OF THE DA |  |  |
|  |  |  |  |
|  |  |  |  |

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

Block diagram with pins used in the Atmega328P

DC Motor

Analog Servo- HD 3001HB

VEXTA Stepper Motor PX243-02AA

1. **INITIAL/DEVELOPED CODE IN C OF TASK 1**

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

volatile unsigned int pushed = 0; // if the motor is on or off

volatile unsigned int ADCvalue;

// Initiate ADC

void ADCinit(void)

{

ADMUX|=(1<<REFS0);

ADCSRA|=(1<<ADSC)|(1<<ADEN)|(1<<ADIE)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE); // start conversion | enable adc| enable interrupts | select 128 prescaler | auto trigger enable | ADC0 used

ADCSRB=0; //initialize ADC in free running mode

TCCR0A|=(1<<COM0A1)|(1<<WGM01)|(1<<WGM00); // the use timer0 in fast PWM mode and set OC0A on bottom then to clear on up counting

TCCR0B|=(1<<CS00); //set the prescaler of 1 to be selected

}

// Interrupt Potentiometer

ISR (ADC\_vect)

{

ADCvalue=ADC; //reads all 16-bits of ADCH:ADCL (0-1023 from pot)

ADCvalue=ADCvalue/4; //divide ADC value by 4 to get 0-225 value

OCR0A=ADCvalue; //set this value to the OCR0A register

*\_delay\_ms*(10); //delay to allow OCR0A to hold

}

// Interrupt Push Button

ISR (INT0\_vect)

{

if(pushed == 0){

pushed = 1;

}

else{

OCR0A = 0;

pushed = 0;

}

*\_delay\_ms*(10);

}

int main()

{

DDRC = 0x00; //make PORTC input pins

DDRD = 0xFB; //PIND2 as Input and the rest of PORTD as output

ADCinit(); //initialize 328 (ADC/timers)

sei(); //enable global interrupts

while(1);

return 0;

}

1. **INITIAL/DEVELOPED CODE IN C OF TASK 2**

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

volatile unsigned int ADCvalue;

// CTC delay subroutine

void CTCdelay(unsigned int adc)

{

TCCR1B |= (1<<WGM13)|(1<<WGM12)|(1<<CS10); // CTC mode | set prescaler to 1

TCNT1 = 0; // intialize counter

OCR1A = adc; // compare value

}

// Initialize ADC

void ADCinit(void)

{

ADMUX|=(1<<REFS0);

//start conversion | enable adc | enable interrupts | select 128 prescaler | auto trigger enable

ADCSRA|=(1<<ADSC)|(1<<ADEN)|(1<<ADIE)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE);

ADCSRB=0; //set adc in free running mode

}

// interrupt ADC

ISR (ADC\_vect)

{

ADCvalue=ADC; //reads all 16-bits of ADCH:ADCL (0-1023 from pot)

PORTB = 0x66; //first step to be outputted

CTCdelay(ADCvalue); // Delay for ADC value

PORTB = 0xCC; //second step to be outputted

CTCdelay(ADCvalue); // Delay for ADC value

PORTB = 0x99; //third step to be outputted

CTCdelay(ADCvalue); // Delay for ADC value

PORTB = 0x33; // fourth step to be outputted

CTCdelay(ADCvalue); // Delay for ADC value

}

int main()

{

DDRC = 0x00; //make PORTC input pins

DDRD = 0xFF; //make PORTD output pin

ADCinit(); //initialize adc

sei(); //enable global interrupts

while(1);

return 0;

}

1. **INITIAL/DEVELOPED CODE IN C OF TASK 3**

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

volatile unsigned int ADCvalue;

// Initialize ADC

void ADCinit(void)

{

ADMUX |= (1<<REFS0);

//start conversion | enable adc | enable interrupts | select 128 prescaler | auto trigger enable

ADCSRA |= (1<<ADSC)|(1<<ADEN)|(1<<ADIE)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE);

ADCSRB = 0; // set adc in free running mode

TCCR1A |= (1<<COM1A1)|(1<<WGM11); // Set timer1 in fast PWM mode | OCR1A clear on up and set down

TCCR1B |= (1<<WGM13)|(1<<WGM12)|(1<<CS11); //prescaler of 1| fast pwm

ICR1 = 19000; //set to 19000 (50Mhz)

}

// Interrupt ADC

ISR (ADC\_vect)

{

ADCvalue=ADC; //reads all 16-bits of ADCH:ADCL (0-1023 from pot)

if(ADCvalue == 0){

OCR1A = 1000; // 0 Degree

}

else if(ADCvalue == 5){

OCR1A = 2000; // 180 Degree

}

*\_delay\_ms*(10);

}

int main()

{

DDRC = 0x00; //make PORTC input pins

DDRB = 0xFF; //make PORTB output

ADCinit(); //initizlize ADC

sei(); //set global interrupts

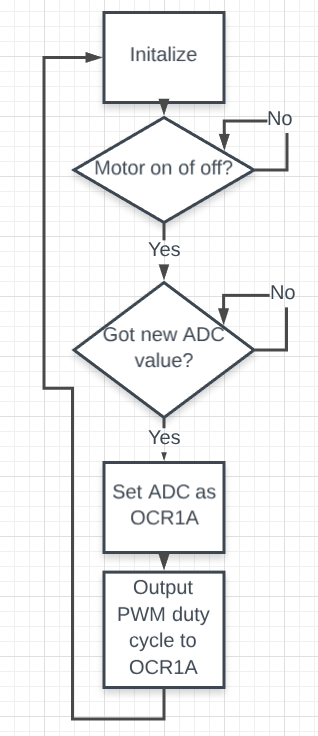
while(1);

return 0;

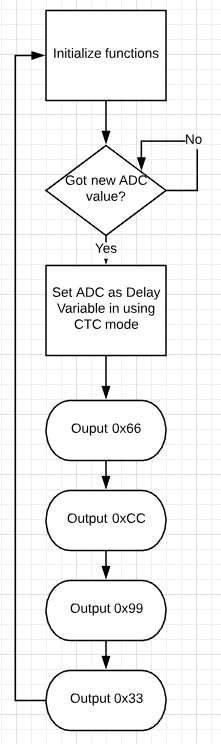
}

1. **FLOWCHART OF EACH TASK**

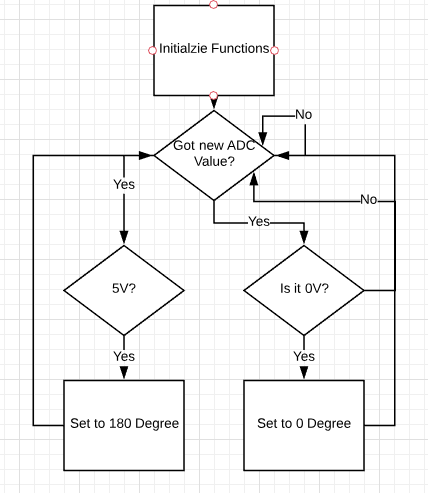
DC MOTOR



STEPPER MOTOR



SERVO MOTOR



1. **SCEENSHOT OF EACH TASK SCHEMATIC**
2. **SCREENSHOT OF EACH TASK ON BREADBOARD**
3. **GITHUB LINK OF DA4**

https://github.com/escalaa/DA04

**Student Academic Misconduct Policy**

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

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

Audie Escala