CPE301 – SPRING 2019

Design Assignment 4A

Student Name: David Flores

Student #: 2000881655

Student Email: flored6@unlv.nevada.edu

Primary Github address: https://github.com/David-Floress/submission\_da.git

Directory: https://github.com/David-Floress/submission\_da

Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

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

List of Components used

ATmega328P

Multi-Function Shield

Bread Board

UNLV ECE Lab Board (Power Source)

Wires

Block diagram with pins used in the Atmega328P

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

/\*

\* DA4A.c

\*

\* Created: 4/12/2019 3:17:45 PM

\* Author : David Flores

\*/

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

volatile unsigned int ADCVal; // Holds ADC Register value

char motor = 0; // Holds the state of the motor (ON/OFF)

void timer\_int (void); // Function declaration for timer

void ADC\_int (void); // Function declaration for ADC

int main(void){

timer\_int(); // Initializes timer

ADC\_int(); // Initializes ADC

while (1)

{

}

}

ISR(PCINT1\_vect)

{

if(!(PINC & (1 << PINC1))) //Button pressed

{

*\_delay\_ms*(50); // Debounce button

while(!(PINC & (1 << PINC1))); // Checks if button was pressed

if(motor) //Start the motor

{

PORTB |= (1 << PORTB1);

// Turns on the motor

ADCSRA |= (1 << ADSC);

// Starts conversion

while((ADCSRA&(1<<ADIF))==0);

// Waits for conversion

ADCVal = ADC & 0x03FF;

// Takes the right 10-bits of ADC register

OCR1A = 10\*ADCVal;

// OCR1A value for duty cycle

}

else if(!motor)

//Stop the motor

{

OCR1A = 0;

// Reset register

PORTB &= ~(1 << PORTB1);

// Turns off motor

}

motor ^= 1; // Remembers button state

}

}

void timer\_int(void)

{

//Port directions

DDRB = (1<<1);

// PB.1-2 (OC1A & OC1B) as output to generate PWM

DDRB |= (1<<5);

// PB.1-2 (OC1A & OC1B) as output to generate PWM

DDRC = 0;

// PORTC as input

PORTB = 0;

// Initially turns off all PB5:0

//Timer1

TCCR1A |= (1 << COM1A1) | (1 << COM1B1) |

(1 << WGM11);

// enable PWM for OC1A & OC1B, Fast PWM, Non-inverted mode

TCCR1B |= (1 << WGM13) | (1 << WGM12) |

(1 << CS11);

// Prescaler = 8

ICR1 = 9999;

// Timer1 TOP

//Interrupt Settings

PORTC |= (1 << PORTC1);

PCICR = (1 << PCIE1);

PCMSK1 = (1 << PCINT9);

sei(); // Enable global interrupts

}

void ADC\_int (void)

{

// ADC initialization

DIDR0 = 0x1;

// disable digital input on ADC0 pin

ADMUX = (1<<REFS0);

// Reference = Aref, ADC0 (PC.0) used as analog input

// right-justified data

ADCSRA |= (1<<ADEN) | (1<<ADPS2) |

(1<<ADPS1) | (1<<ADPS0);

// enable ADC, system clock used for ADC

ADCSRB = 0x0;

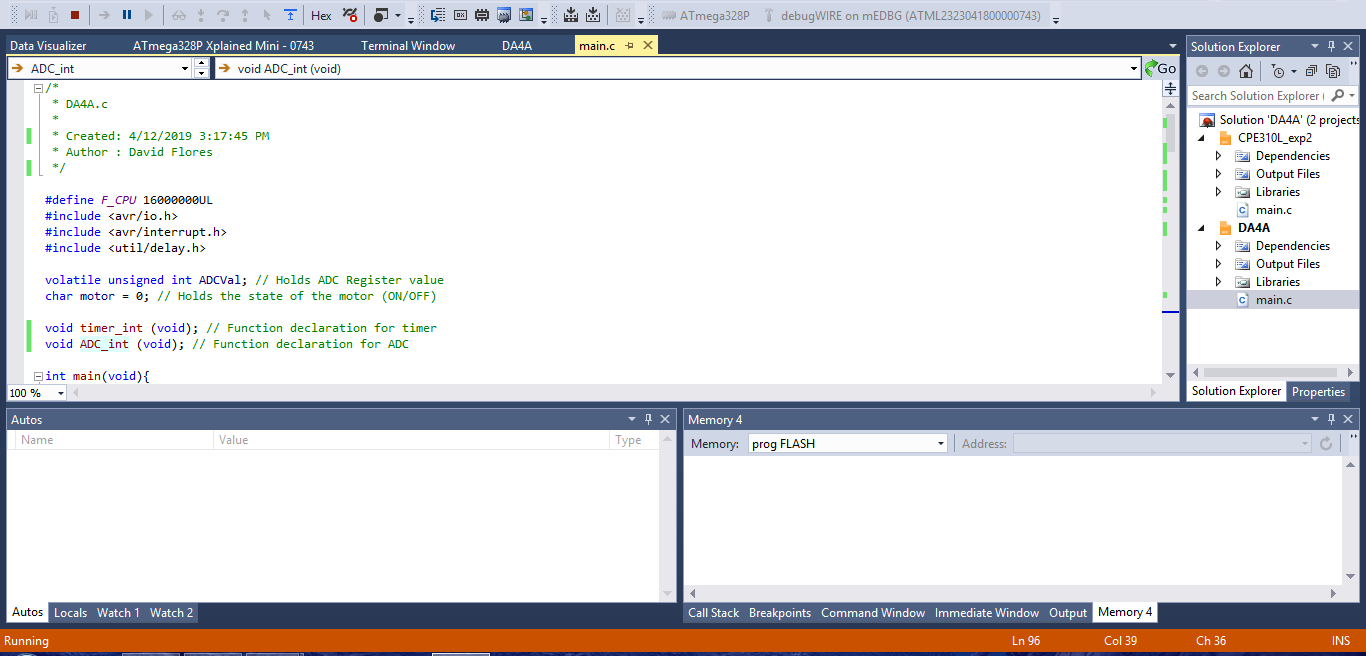
// free running mode

}

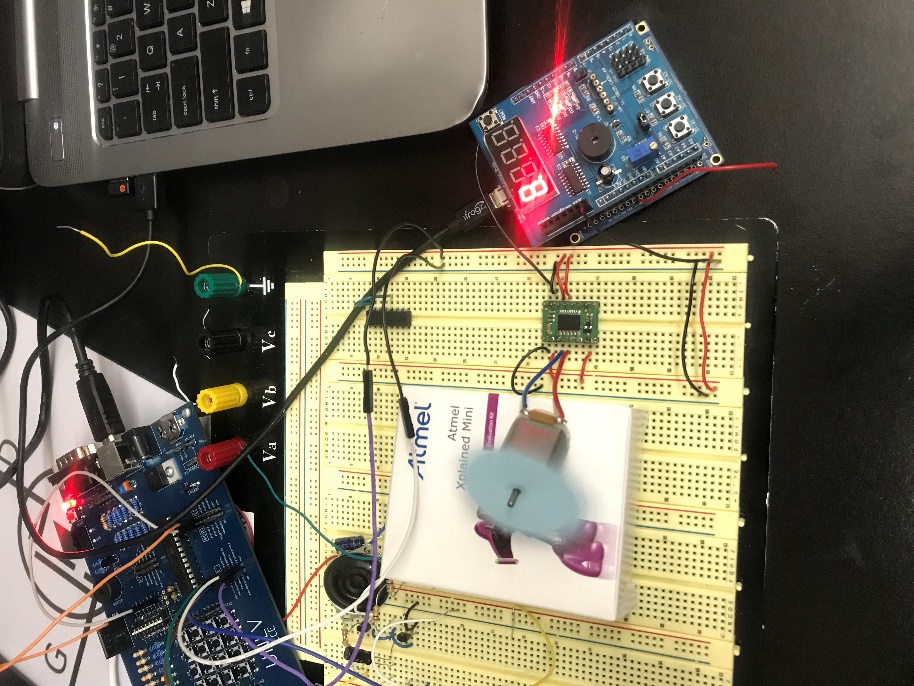
1. **DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A**

**N/A**

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



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



1. **VIDEO LINKS OF EACH DEMO**

<https://www.youtube.com/watch?v=v_qmMd-96YY>

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”.

NAME OF THE STUDENT