CPE301 – SPRING 2019

Design Assignment 2A

Student Name: Allis Hierholzer

Student #: 2000160356

Student Email: hierholz@unlv.nevada.edu

Primary Github address: https://github.com/acexhp/submission\_da.git

Directory: Repository/cpe301/DesignAssignment/DA2A

Task:

1. Design a delay subroutine to generate a waveform on PORTB.2 with 60% DC and 0.725 sec period.

2. Connect a switch to PORTC.2 (active high - turn on the pull up transistor) to poll for an event to turn on the led at PORTB.2 for 1.250 sec after the event.

Submission:

The following are required for successful completion of the design assignment:

a. AVR ASM code that has been compiled and working for all tasks. Verify the period and duty cycle of the waveforms in simulation and emulation.

b. AVR C code that has been compiled and working for all tasks. Verify the period and duty cycle of the waveforms in simulation and emulation.

c. The C code should be well documented with explanation of every instruction.

d. A word document that contains the code with comments, complete schematics, that includes the AVR, components connected on the breadboard and LED should be included. Follow the template provided.

e. A snapshot of the board with connected components and a video of the complete LED bar blink sequence should be recorded and uploaded to Youtube and the line to be provided for each task.

f. The git directory should have DA2\DA2T1, DA2\DA2T2, … folders, with one doc file and video link file.

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

* ATMEGA328P XPLAINED MINI
* MULTIFUNCTION SHIELD
* OSCILLOSCOPE
* PROBES

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

**TASK 1A: ASSEMBLY CODE**

;

; 1assembly.asm

; Author : hierholz

;

.org 0

LDI R16,4 ;representing PB2

OUT DDRB, R16 ;enable PB2 as output

LDI R17,0 ;resets PB2

LDI R20,5 ;set clock prescaler to 1024

STS TCCR1B,R20

begin:

LDI R20,0x00 ;reset counter to 0

STS TCNT1H,R20 ;reset high bit to 0

STS TCNT1L,R20 ;reset low bit to 0

RCALL delay ;calls label delay

OUT PORTB,R17 ;LED turns on for 435ms

EOR R17,R16 ;XOR to toggle led

LDI R20,0x00 ;resetting the counter to 0

STS TCNT1H,R20 ;resetting high bit to 0

STS TCNT1L,R20 ;resetting low bit to 0

RCALL delay2 ;calls label delay2

OUT PORTB,R17 ;LED turns on for 290ms

EOR R17,R16 ;XOR to toggle led

RJMP begin ;repeat main loop

delay:

LDS R29, TCNT1H ;loading upper bit of counter to R29

LDS R28, TCNT1L ;loading lower bit of counter to R28

CPI R28,0x8C ;comparing if lower 8 bits of timer is 0x8C

BRSH body ;if lower bits of timer reached desired amount,

Checking upper bits

RJMP delay ;otherwise, keep checking lower bits

body:

CPI R29,0x1A ;check to see if upper timer bits have reached the

desired value

BRLT delay ;if not, recheck the lower bits

RET ;once the timer reached the desired value, toggle the

LED

delay2:

LDS R29, TCNT1H ;loading upper bit of counter to R29

LDS R28, TCNT1L ;loading lower bit of counter to R28

CPI R28,0xB2 ;comparing if lower 8 bits of timer is 0xB2

BRSH body2 ;if lower bits of timer have reached desired amount,

check the upper bits

RJMP delay2 ;otherwise, keep checking lower bits

body2:

CPI R29,0x11 ;check to see if upper timer bits have reached the

desired value

BRLT delay2 ;if not, recheck the lower bits

RET ;once the timer reached the desired value, toggle the

LED

**TASK 1B: C CODE**

/\*

\* 1ccode.c

\* Author : hierholz

\*/

#define *F\_CPU* 16000000UL /\* clock runs at 16 MHz\*/

#include <avr/io.h>

#include <util/delay.h> /\* library that is needed to be able to use

certain functions\*/

int main(void)

{

DDRB = 0x04; /\* pin 2 in port B an output \*/

while(1)

{

PORTB = (1 << PORTB2); /\* displays output \*/

*\_delay\_ms*(435); /\* time that the led is off (60% DC with 0.725s

period) \*/

PORTB = ~(1 << PORTB2); /\* displays output \*/

*\_delay\_ms*(290); /\* time that the led is on (40% DC with 0.725s

period) \*/

}

return 0;

}

**TASK 2A: ASSEMBLY**

;

; 2assembly.asm

; Author : hierholz

;

.org 0

.include <m328pdef.inc> ;defining chip

SBI DDRB, 5 ;turns off LED D1

SBI PORTB, 5

SBI DDRB, 2 ;PB2 an output

CBI DDRC, 2 ;PC2 an input

AGAIN:

SBIS PINC, 2 ;skips the next instruction if PC2 is high

RJMP OVER ;jump to label OVER

SBI PORTB, 2

RJMP AGAIN ;jump to label AGAIN

OVER:

CBI PORTB, 2 ; make port B2 high

LDI R20,5 ;set clock prescaler to 1024

STS TCCR1B,R20

LDI R20,0x00 ;reset counter to 0

STS TCNT1H,R20 ;reset high bit to 0

STS TCNT1L,R20 ;reset low bit to 0

delay:

LDS R29, TCNT1H ;loading upper bit of counter to R29

LDS R28, TCNT1L ;loading lower bit of counter to R28

CPI R28,0x4A ;comparing if lower 8 bits of timer is 0x4A

BRSH body ;if lower bits of timer reached desired amount,

;Checking upper bits

RJMP delay ;otherwise, keep checking lower bits

body:

CPI R29,0x4C ;check to see if upper timer bits have reached the

;desired value

BRLT delay ;if not, recheck the lower bits

RET ;once the timer reached the desired value, toggle the

;LED

**TASK 2B: C CODE**

/\*

\* 2ccode.c

\* Author : hierholz

\*/

#define *F\_CPU* 16000000UL /\* clock runs at 16 MHz\*/

#include <avr/io.h>

#include <util/delay.h> /\* library that is needed to be able to use

certain functions\*/

int main(void)

{

DDRB |= (1<<2); /\* set PORTB2 as output \*/

PORTB |= (1<<2); /\* Turn LED off \*/

DDRC &= (0<<2); /\* set PORTC1 for input \*/

PORTC |= (1<<2); /\* enable pull-up \*/

/\* when the PINC is pressed, LED pulses for 1.25s \*/

while (1) {

if (!(PINC & (1<<PINC1)))

{

PORTB &= ~(1<<2);

*\_delay\_ms*(1250);

}

else

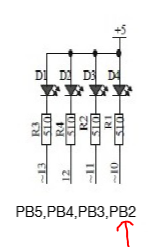
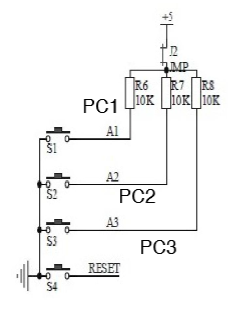
PORTB |= (1<<2);

}

return 0;

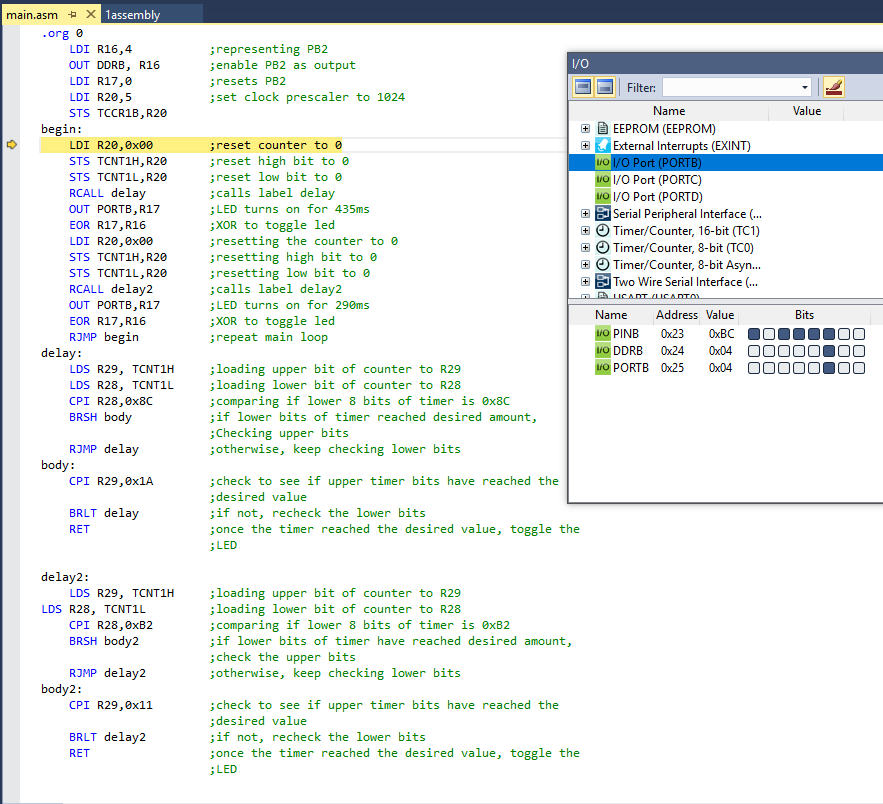
}

1. **SCHEMATICS**

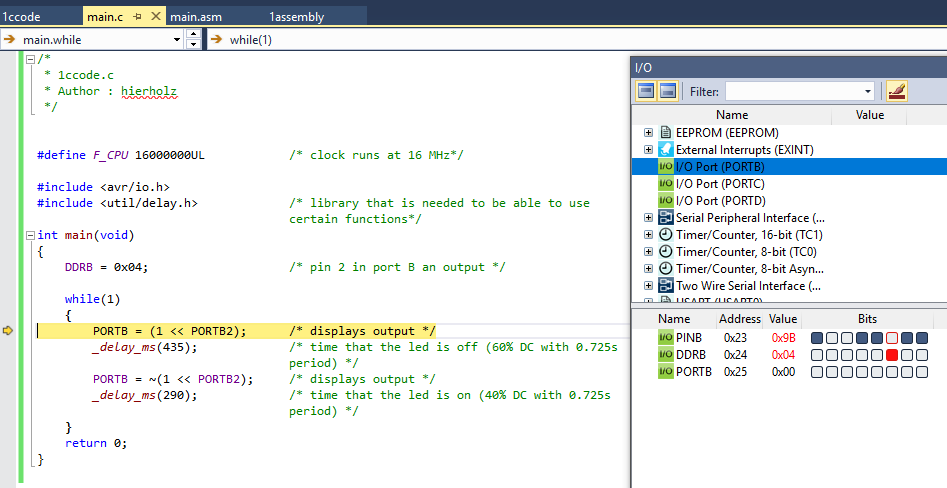


1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

**TASK 1A: ASSEMBLY CODE**

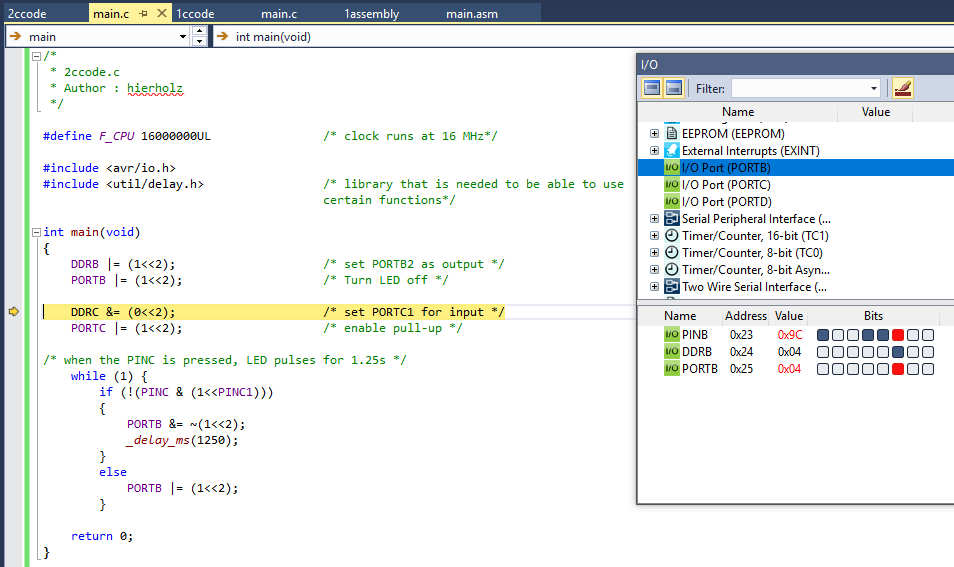


**TASK 1B: C CODE**



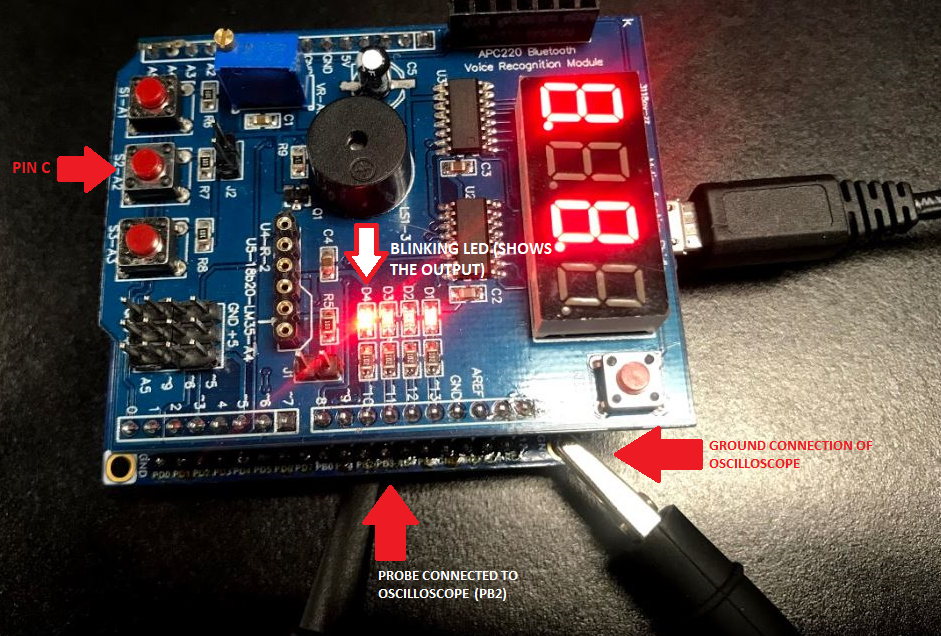
**TASK 2A: ASSEMBLY**

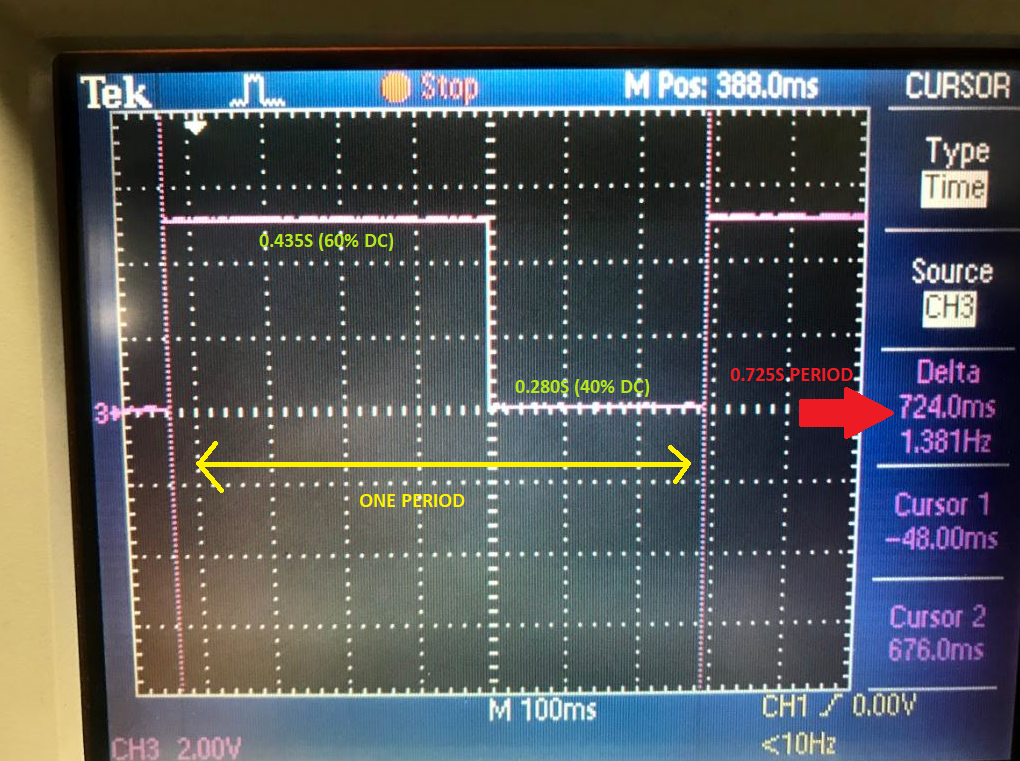
**TASK 2B: C CODE**



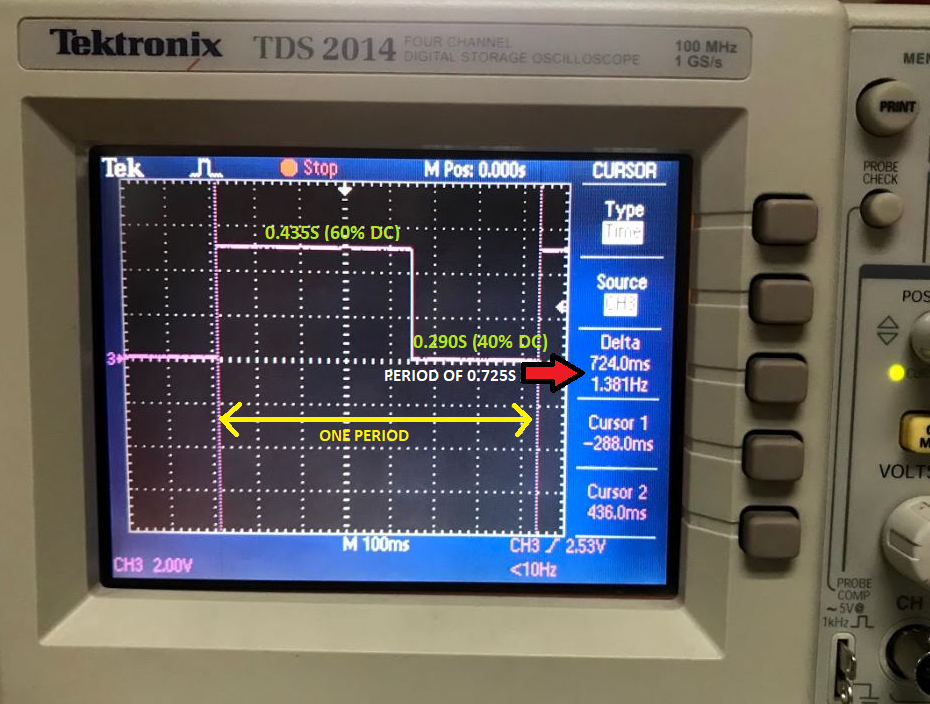
1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

**The board set up is the same for all exercises; the only different is for exercise 2, it requires an input push button on PIN C**

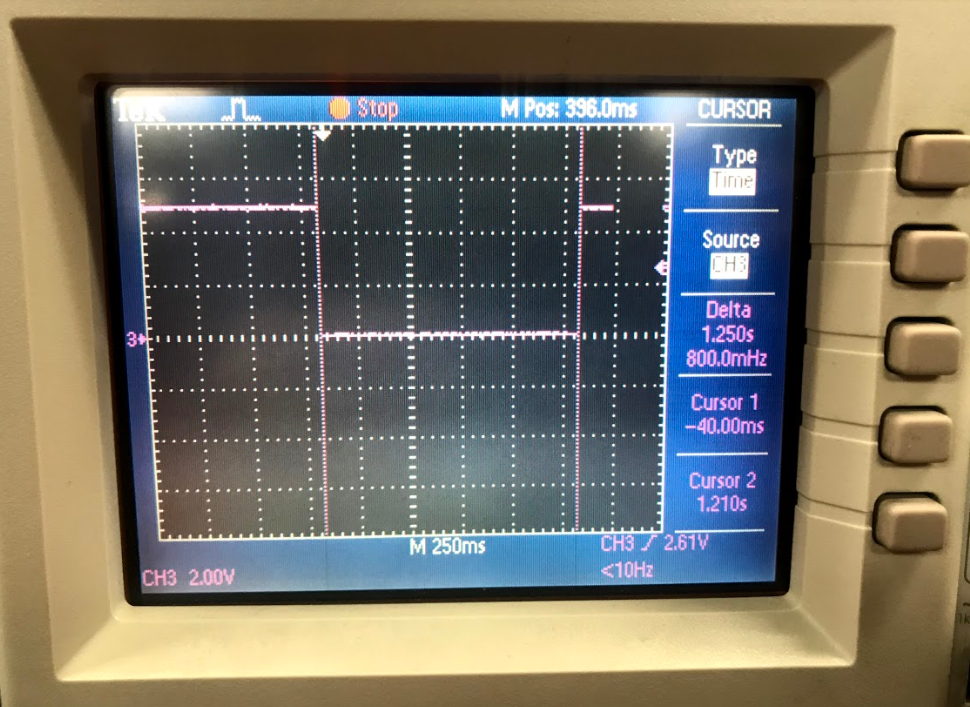


**TASK 1A: ASSEMBLY CODE**

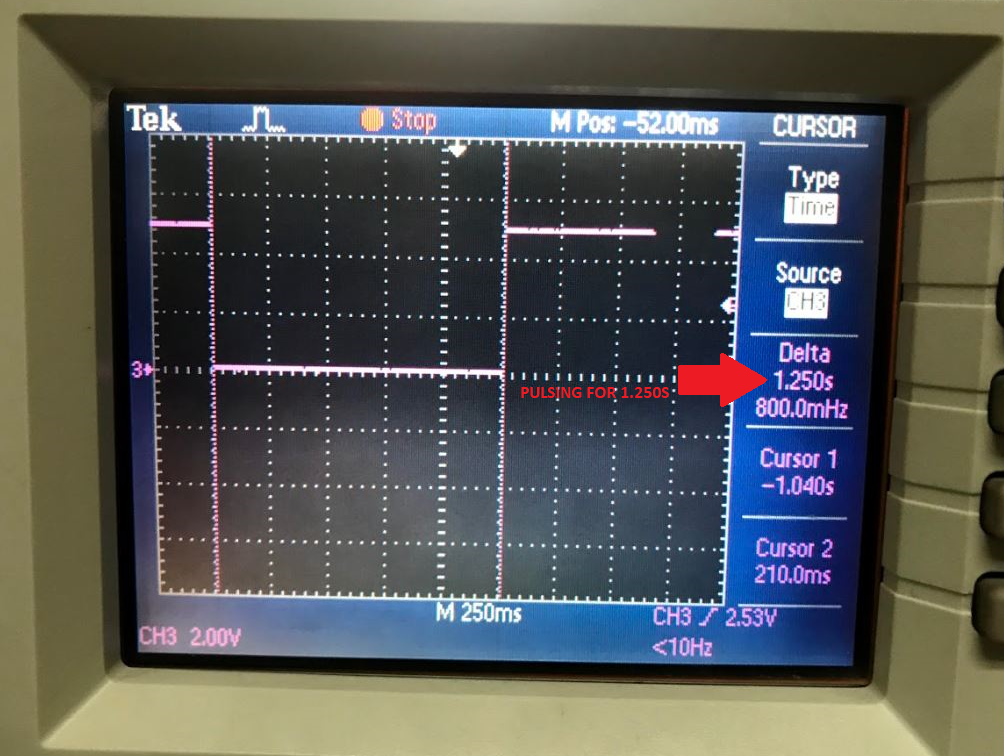
**TASK 1B: C CODE**



**TASK 2A: ASSEMBLY**



**TASK 2B: C CODE**



1. **VIDEO LINKS OF EACH DEMO**

**TASK 1A: ASSEMBLY**

<https://youtu.be/O4cYKouQ3vo>

**TASK 2B: C CODE**

<https://youtu.be/E_To1gDcMI4>

**TASK 2A: ASSEMBLY**

<https://youtu.be/VJjxDn8Oeak>

**TASK 2B: C CODE**

<https://youtu.be/Ooltkr0D4qk>

1. **GITHUB LINK OF THIS DA**

<https://github.com/acexhp/submission_da.git>

**Student Academic Misconduct Policy**

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

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

Allis Hierholzer