CPE301 – SPRING 2022

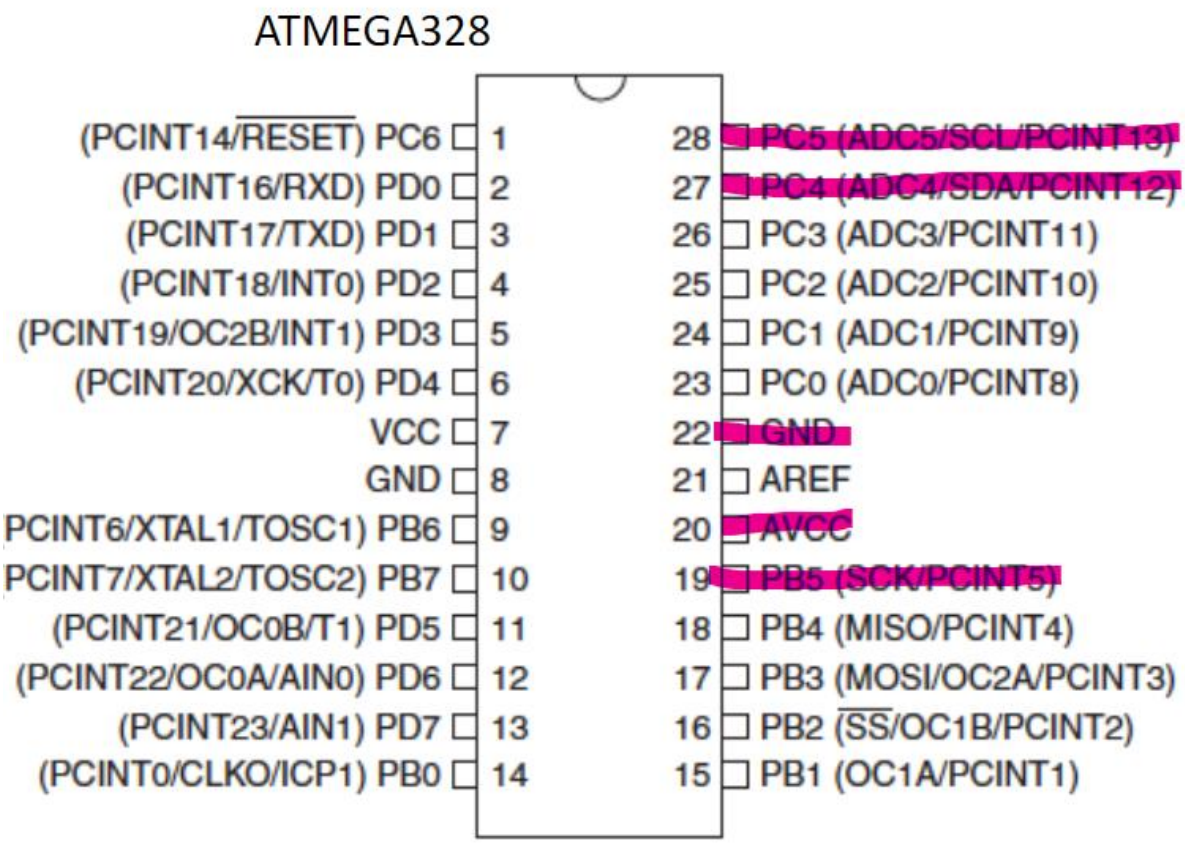
03 Analog to Digital Conversion

Student Name: [redacted]
Student #: [redacted]
Student Email: [redacted]
Primary Github address: https://github.com/DylanCaz/Submission_DA.git
Directory:
https://github.com/DylanCaz/Submission_DA/tree/main/Design_Assignments_sub/DA_3_sub

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



2. INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/2

```
#define F_CPU 16000000UL
//#define UBRR_9600 103 // for 16Mhz with .2% error
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>
#define BAUDRATE 9600
#define BAUD_PRESCALLER (((F_CPU / (BAUDRATE * 16UL))) - 1)

uint16_t adcValueX; // Variable used to store read value of ADC
uint16_t adcValueY; // Variable used to store read value of ADC
uint8_t i = 0;
char buffer[5];     //Output of the itoa function

// Function Declarations
void adc_init(void); //Function to initialize/configure ADC
uint16_t read_adc(uint8_t ADCchannel); // Function to read channel
void USART_init(void); //Function to initialize and configure USART
```

```
void USART_send(unsigned char data); // Function to send char to serial port
void USART_tx_string( char *stringPtr ); // Function to send string to serial port
//void blink(void);
void led_blink(void);


int main()
{
	adc_init(); // Start ADC
	USART_init(); // Start USART
	USART_tx_string("Connected! \r\n"); // We're alive!
	_delay_ms(125); // wait a bit
	USART_tx_string(buffer);
	//led();


	while(1)
	{
		adcValueX = read_adc(4);  // Read ADC value at PC4 channel
		USART_tx_string("X-Axis: ");
		itoa(adcValueX, buffer, 10);
		USART_tx_string(buffer);
		USART_tx_string(" , ");

		USART_tx_string("Y-Axis: ");
		adcValueY = read_adc(5); // Read ADC value at PC5 channel
		itoa(adcValueY, buffer, 10);
		USART_tx_string(buffer);
		//USART_tx_string(",");
		_delay_ms(500);
		USART_send('\n');
		//blink();
		led_blink();
	}
}



//Function to initialize/configure ADC
void adc_init(void)
{
	ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); // ADC_CLK = 16MHz/ 128 =
125kHz
	ADMUX |= (1 << REFS0); // AVcc = 5V
	ADCSRA |= (1 << ADEN); // enables ADC
	ADCSRA |= (1 << ADSC); // starting ADC conversion
}
// Function to read channel
uint16_t read_adc(uint8_t ADCchannel)
{
	ADMUX = (ADMUX & 0xF0) | (ADCchannel & 0x0F); // select ADC with safety mask
	ADCSRA |= (1 << ADSC); // Starting new conversion
	while(ADCSRA & (1 << ADSC));	// Waiting until conversion is complete
	return ADCW; // Return ADC value of chosen channel
}
//Function to initialize and configure USART
void USART_init(void)
{
```

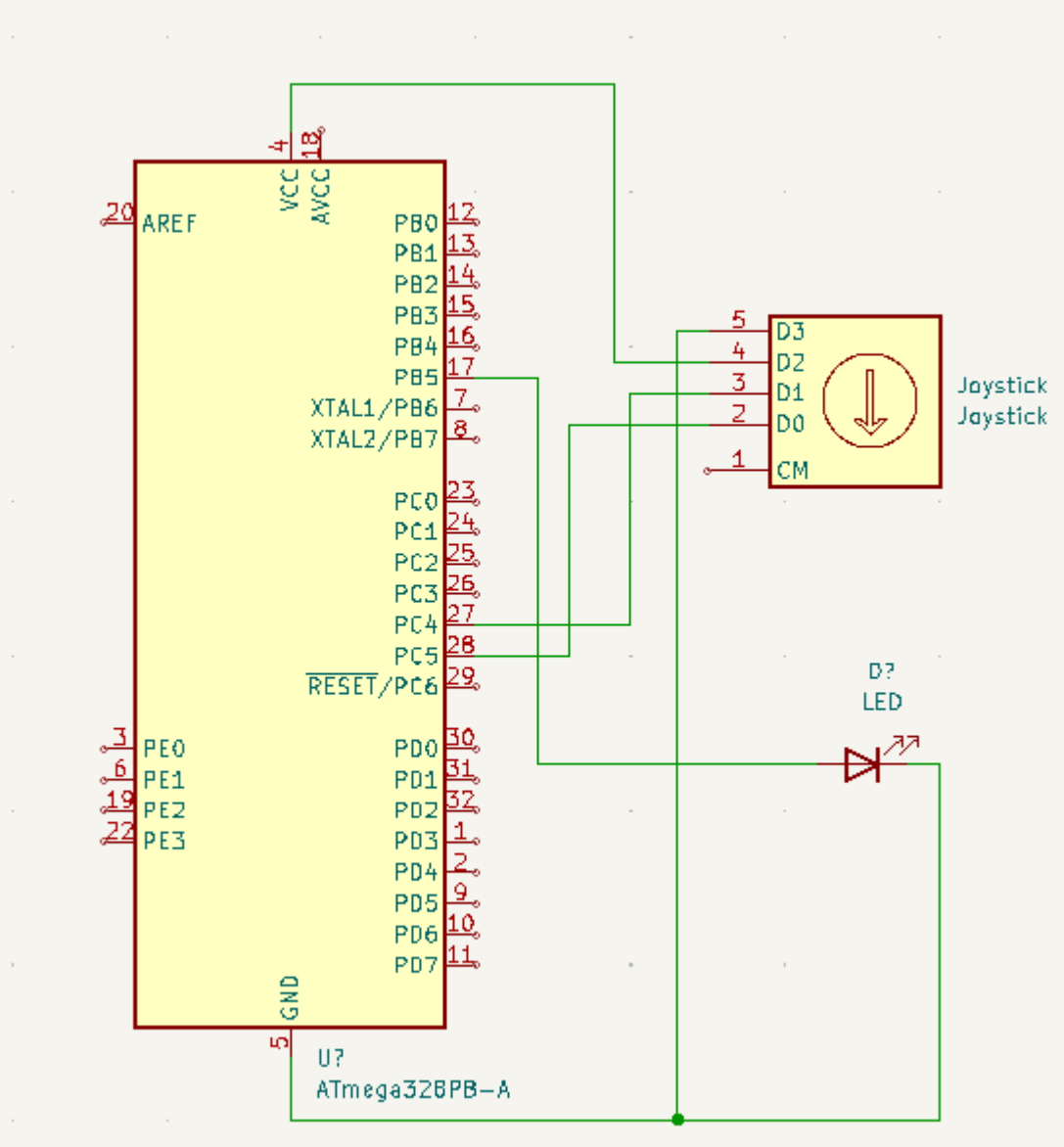
```
	UBRR0H = (uint8_t)(BAUD_PRESCALLER >> 8);
	UBRR0L = (uint8_t)(BAUD_PRESCALLER);
	UCSR0B = (1 << TXEN0) | (1 << RXEN0); // enable transmission and reception
	UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // set frame formate to 8bits, no parity,
1
}
void USART_send(unsigned char data)
{
	while (!(UCSR0A & (1 <<UDRE0)));
	UDR0 = data;
}
// Function to send string to serial port
void USART_tx_string( char *stringPtr )
{
	while ((*stringPtr != '\0'))
	{
		USART_send(*stringPtr);
		stringPtr++;
	}
}
```

3. DEVELOPED MODIFIED CODE OF TASK 3

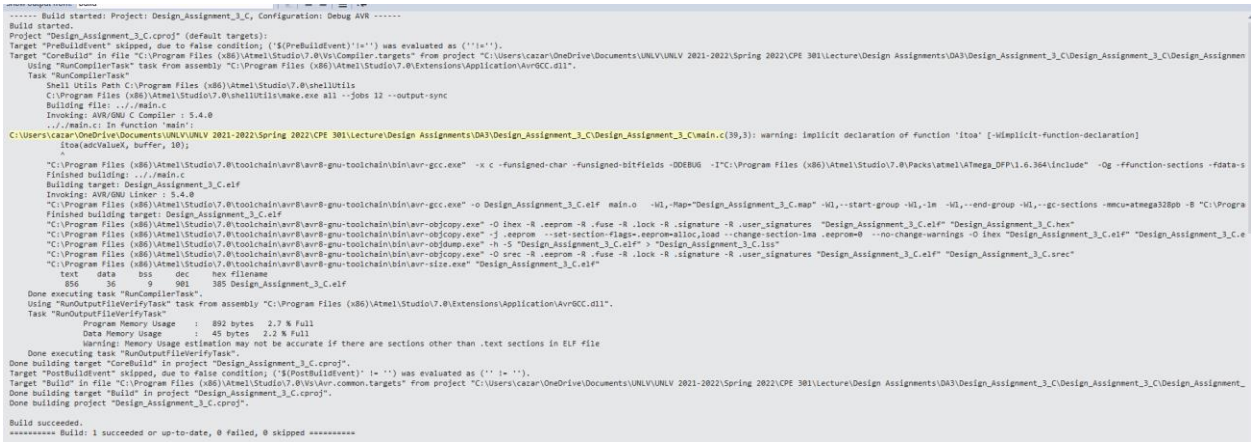
```
// Task 3
void led_blink(void)
{
	DDRB |= (1 << 5);
	//CTC mode, Prescalar = 1024
	TCCR1B = (1 << WGM12) | (1 << CS12) | (1 << CS10);
	TCNT1 = 0;

	if(adcValueX < 498)
	{
		OCR1A = 130; // 60Hz
		while((TIFR1 & (1 << OCF1A)) == 0)
		{
			PORTB ^= (1 << 5);
			TCNT1 = 0;
			TIFR1 = (1 << OCF1A);
		}
	}
	else if(adcValueX > 498)
	{
		//OCR1A = 130 - (1023-adcValueX)/4; // 60Hz
		OCR1A = 7813; // 1Hz
		while((TIFR1 & (1 << OCF1A)) == 0);
		{
			PORTB ^= (1 << 5);
			TCNT1 = 0;
			TIFR1 = (1 << OCF1A);
		}
	}

	if (adcValueX == 498)
	{
```

```
                OCR1A = 260; // 30Hz
                while((TIFR1 & (1 << OCF1A)) == 0);
                {
                        PORTB ^= (1 << 5);
                        TCNT1 = 0;
                        TIFR1 = (1 << OCF1A);
                }

        }
}
```

4. SCHEMATICS

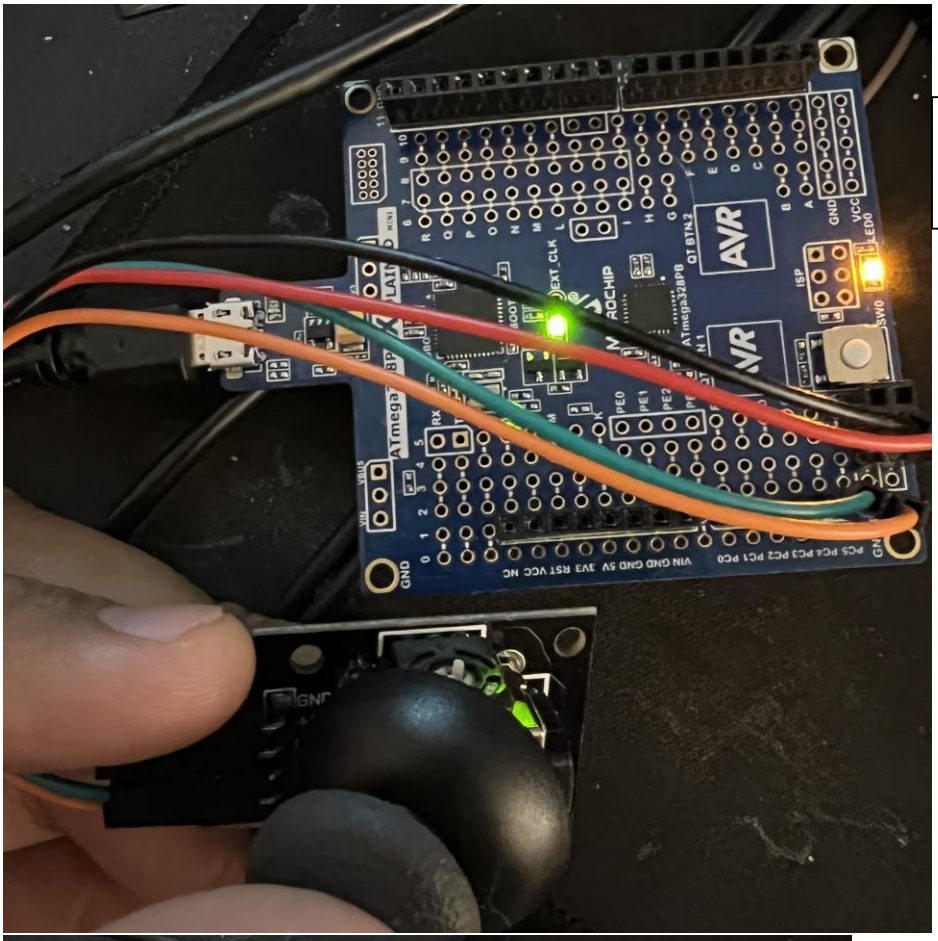


5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

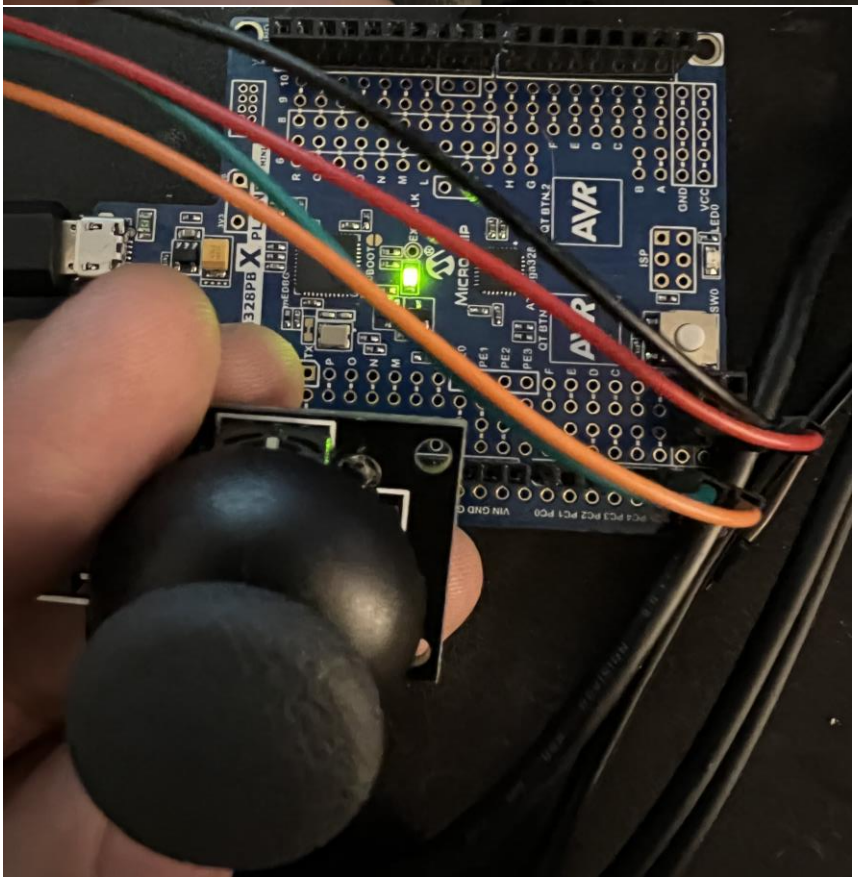


6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

```
X-Axis: 498 , Y-Axis: 496
X-Axis: 498 , Y-Axis: 496
X-Axis: 497 , Y-Axis: 479
X-Axis: 497 , Y-Axis: 497
X-Axis: 1023 , Y-Axis: 485
X-Axis: 1023 , Y-Axis: 476
X-Axis: 496 , Y-Axis: 493
X-Axis: 0 , Y-Axis: 495
X-Axis: 0 , Y-Axis: 494
X-Axis: 497 , Y-Axis: 497
X-Axis: 498 , Y-Axis: 497
X-Axis: 497 , Y-Axis: 497
X-Axis: 497 , Y-Axis: 497
```



LED Blinking on and off,
Joystick to the left
Joystick to the center



7. VIDEO LINKS OF EACH DEMO

https://www.youtube.com/watch?v=QvE4CocJwiY

8. GITHUB LINK OF THIS DA

https://github.com/DylanCaz/Submission_DA/tree/main/Design_Assignments_sub/DA_3_sub

Student Academic Misconduct Policy

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

"*This assignment submission is my own, original work*".

Dylan Cazares