Design Assignment 4

Student Name: Samuel McCormick
Student #: 1014303276
Student Email: samuel.mccormick@gmail.com
Primary Github address: https://github.com/brokenboredom/tech-muffin.git
Directory: DesignAssignments/DA4

Program the UART Module to interact with the ATmega328pb:
1. On transmitting the following keys from the host terminal, the following actions will be performed:

Code sections dealing with USART interrupts

```
//Important Variables
char outs[20];
char begin[] = "Begin Program\r\n";
//char INVALID_INPUT[] = "Invalid Input\n";
char HELP0[] = "\nUsage:\r\n";
char HELP1[] = "- h Displays this screen\r\n";
char HELP2[] = "- o turns ON LED at PB5\r\n";
char HELP3[] = "- O turns OFF LED at PB5\r\n";
char HELP4[] = "- b Blink LED at PB3\r\n";
char HELP5[] = "- P stop blinking LED at PB3\r\n";
char HELP6[] = "- a read ADC value from POT\r\n";
char HELP7[] = "- A stop reading ADC values\r\n";


/* Initializes the USART (RS232 interface) */
void USART_init(unsigned int ubrr) {
        UBRR0H = (ubrr >> 8);
        UBRR0L = (ubrr);
        UCSR0B = (1 << TXEN0) | (1 << RXEN0) | (1 << RXCIE0); // Enable receiver, transmitter & RX interrupt
        //asynchronous 8 N 1 // mega328
        UCSR0C = (0 << UMSEL01) |
        (0 << UMSEL00) | // 00 async operation, 01 synch operation
        (0 << UPM01) | // Parity - 0 Disabled, 0 Reserved, 1 Enabled Even, 1 Enabled Odd
        (0 << UPM00) | // Parity - 0 Disabled, 1 Reserved, 0 Enabled Even, 1 Enabled Odd
        (0 << USBS0) | // stop Bits - 0 = 1bit 1 = 2bit
        (1 << UCSZ01) | // 8 Data bits
        (1 << UCSZ00) | //
        (0 << UCPOL0); // for Synch Mode only - clock polarity
}

/* Send some data to the serial port */
void USART_tx_string( char *data ) {
        while ((*data != '\0')) {
                while (!(UCSR0A & (1 <<UDRE0)));
                UDR0 = *data;
                data++;
        }
}

//Main section
int main(void)
{
```

```
	DDRB |= (1<<PB5) | (1<<PB3); //PB5 and PB3 as output
	PORTB |= (1<<PB5); //Turn off (power) led on Shield

	// Configure Timer2
	TCNT2 = 0;
	OCR2A = 255;
	TCCR2A |= (1<<WGM21) | (0<<WGM20);                              // CTC mode
	TCCR2B |= (0<<WGM22) | (1<<CS22) | (1<<CS21) | (1<<CS20);       // Prescaler to 1024
	TIMSK2 |= (0<<OCIE2A);                                          // Disable timer interrupt (until specified)
	COMPA_count = 0;                                                // Compare match count

	adc_init(); //Initialize ADC
	USART_init(BAUD_PRESCALLER); //Initialize USART0
	USART_tx_string(begin);
	ADCSRA |= (1<<ADSC); //trigger
	_delay_ms(125);
	sei();
	//ADCSRA |= (1<<ADSC); //trigger ADC in free-running mode
	while (1) {

	}
}
```

1. On-reboot or ‘h’ key – help screen (list all keys and functionalities). At any point of code execution ‘h’ should display the help screen.

```
ISR(USART_RX_vect)
{
	unsigned char data = UDR0;
	//usart_send_byte(data);    //echo what we type in the terminal
	//determine what needs to be done based on typed char
	// ‘h’ key – help screen (list all keys and functionalities)
		if (data == 'h'){
			USART_tx_string(HELP0);
			_delay_ms(125);
			USART_tx_string(HELP1);
			_delay_ms(125);
			USART_tx_string(HELP2);
			_delay_ms(125);
			USART_tx_string(HELP3);
			_delay_ms(125);
			USART_tx_string(HELP4);
			_delay_ms(125);
			USART_tx_string(HELP5);
			_delay_ms(125);
			USART_tx_string(HELP6);
			_delay_ms(125);
			USART_tx_string(HELP7);
		else if {...} //other code
}
```

2. ‘o’ - turns ON LED at PB5, ‘O’ turns OFF the LED at PB5.

```
ISR(USART_RX_vect)
{
        unsigned char data = UDR0;
        //usart_send_byte(data);    //echo what we type in the terminal
        //determine what needs to be done based on typed char
        // ‘h’ key – help screen (list all keys and functionalities)
                if (data == 'h'){...}
                else if (data == 'o'){
                        PORTB &= ~(1<<PB5); //Turn on LED
                }
                else if (data == 'O'){
                        PORTB |= (1<<PB5); //Turn off LED
                }

              else if {...}
}
```

3. ‘b’ - Blink (on-off) the LED PB3. Choose your own period and duty cycle for the on-off blinking. Only use timers. ‘P’ turns off the LED PB3/stops this operation.

```
ISR(USART_RX_vect)
{
        unsigned char data = UDR0;
        //usart_send_byte(data);    //echo what we type in the terminal
        //determine what needs to be done based on typed char
        // ‘h’ key – help screen (list all keys and functionalities)
                if {...}
                else if (data == 'b'){
                                TIMSK2 |= (1 << OCIE2A); //Enable interrupt
                }
                else if (data == 'P'){
                                TIMSK2 &= ~(1 << OCIE2A); //Disable interrupt
                }
                else if {...}
}

// Configure Timer2
        TCNT2 = 0;
        OCR2A = 255;
        TCCR2A |= (1<<WGM21) | (0<<WGM20);                              // CTC mode
        TCCR2B |= (0<<WGM22) | (1<<CS22) | (1<<CS21) | (1<<CS20);       // Prescaler to 1024
        TIMSK2 |= (0<<OCIE2A);                                  // Disable timer interrupt (until specified)
        COMPA_count = 0;                                                // Compare match count
```

4. ‘a’ – reads the ADC value from the POT connected to AC0/PC0. Display the ADC value as a voltage level in terminal. Keep displaying the value every 0.10 sec until ‘A’ stops display and requests the next command. Use Timer auto-trigger for this implementation.

```
//ADC initialize
void adc_init(void)
{
	ADMUX |= (0 << REFS1) |  //Voltage reference bits
	(1 << REFS0) |
	(0 << ADLAR) |  //Left adjust
	(0 << MUX2) |   //Source selector
	(0 << MUX1) |
	(0 << MUX0);
	ADCSRA |= (1 << ADEN) | //Enable ADC
	(0 << ADSC) |           //Start conversion
	(1 << ADATE) |          //Enable auto trigger
	(0 << ADIF) |          //Interrupt flag
	(0 << ADIE) |          //Enable interrupt
	(1 << ADPS2) |          //Prescalar
	(0 << ADPS1) |
	(1 << ADPS0);
	ADCSRB |= (0 << ADTS2) |  //Auto-trigger source
	(0 << ADTS1) |
	(0 << ADTS0);
}

//ADC conversion complete ISR
ISR(ADC_vect)
{
	adc_temp += ADC; //sum
	j--;
	if (j==1)
	{
		adc_temp /= 4;
		snprintf(outs,sizeof(outs),"%3d\r\n", adc_temp);
		adc_temp = 0;
		j = 4;
		USART_tx_string(outs);
		_delay_ms(125);
	}
	else {
		//ADCSRA |= (1<<ADSC); //trigger another
	}
}
```

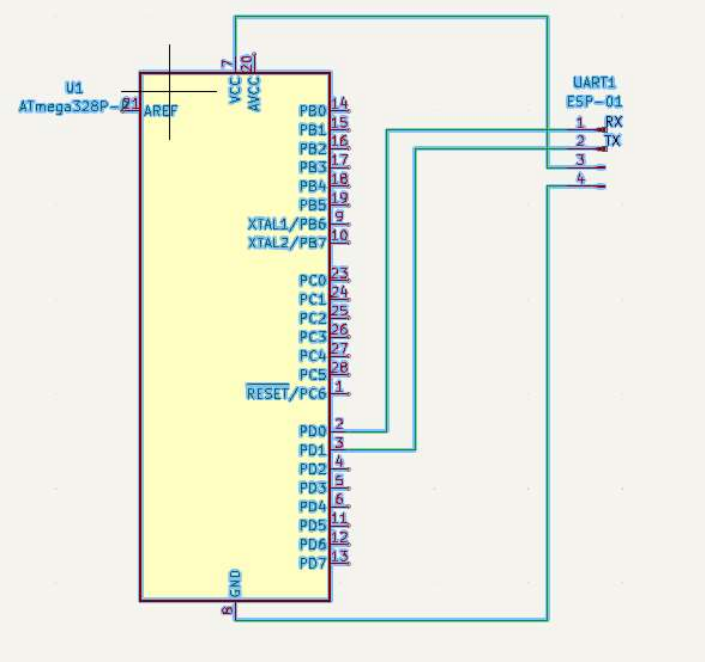
5. Use UART RX interrupt for all of the above operations.

Full USART RX interrupt vector.

```
ISR(USART_RX_vect)
{
	unsigned char data = UDR0;
	//usart_send_byte(data);    //echo what we type in the terminal
	//determine what needs to be done based on typed char
```

```
        // ‘h’ key – help screen (list all keys and functionalities)
                if (data == 'h'){
                        USART_tx_string(HELP0);
                        _delay_ms(125);
                        USART_tx_string(HELP1);
                        _delay_ms(125);
                        USART_tx_string(HELP2);
                        _delay_ms(125);
                        USART_tx_string(HELP3);
                        _delay_ms(125);
                        USART_tx_string(HELP4);
                        _delay_ms(125);
                        USART_tx_string(HELP5);
                        _delay_ms(125);
                        USART_tx_string(HELP6);
                        _delay_ms(125);
                        USART_tx_string(HELP7);
                }
                else if (data == 'o'){
                        PORTB &= ~(1<<PB5); //Turn on LED
                }
                else if (data == 'O'){
                        PORTB |= (1<<PB5); //Turn off LED
                }
                else if (data == 'b'){
                                TIMSK2 |= (1 << OCIE2A); //Enable interrupt
                }
                else if (data == 'P'){
                                TIMSK2 &= ~(1 << OCIE2A); //Disable interrupt
                }
                else if (data == 'a'){
                        //Read ADC working, set to run on "-o" key
                        ADCSRA |= (1 << ADIE); //Turn on interrupt
                }
                else if (data == 'A') {
                        ADCSRA &= ~(1 << ADIE); //Turn off interrupt
                }
}
```

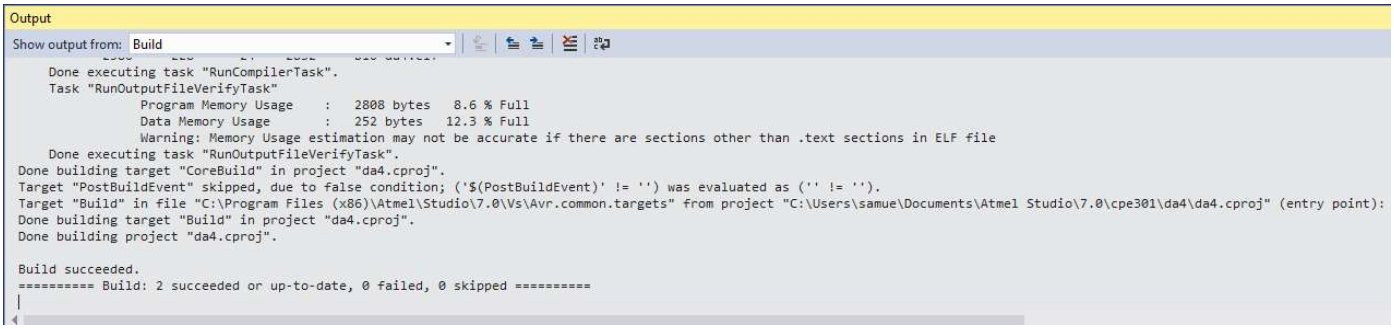
Schematics:



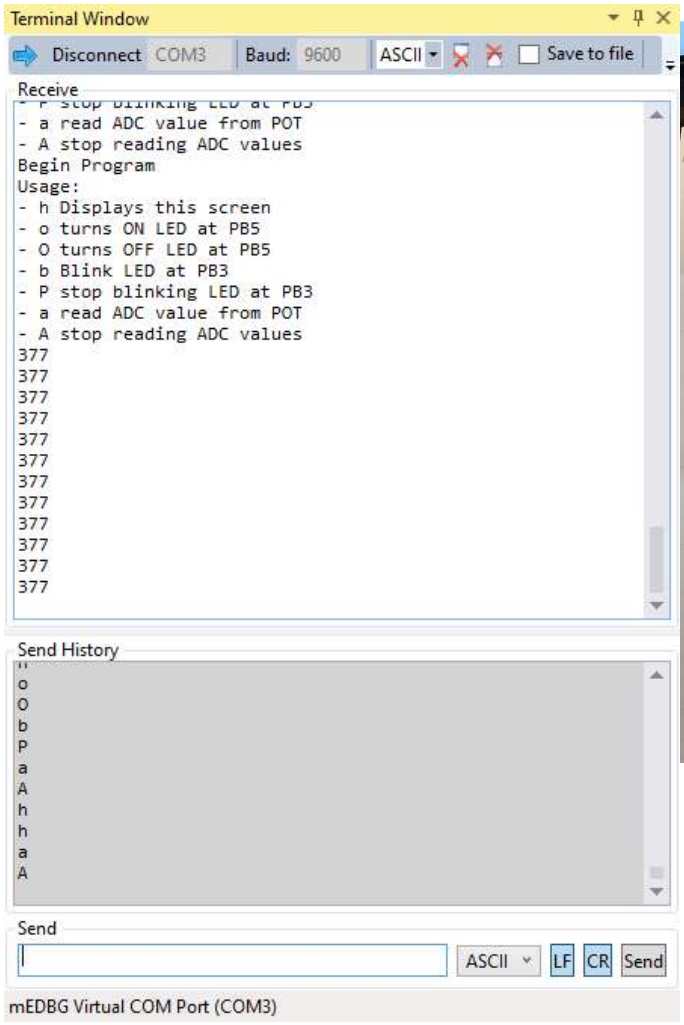
Schematic of ESP-01 module connected to Atmega328p.

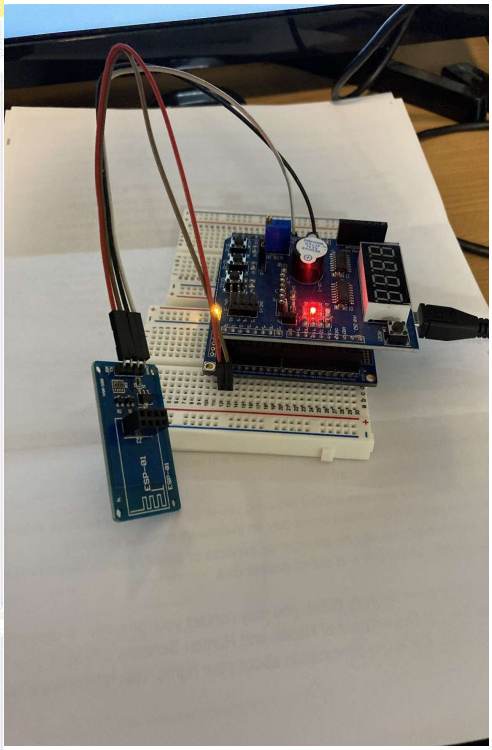
Captures:

Please find screenshots for compilation, demo circuit, and demo output.



Successful compilation.





Demo output terminal.(left)
Demo circuit.(right)

Video Demo Link:
https://youtu.be/akYkL7zekqc