```
/*****************************************************************************
* Project name: main.c
* Created:  11/14/2023 7:17:04 PM
* Author: Caiden Moreno
* Overview:The purpose of this program is to first test the LCD to ensure it operates
* properly throughout the rest of the
* objectives. The second objective of the program is to use USART communication to ensure
* that the interrupts are sending
* and storing information to be sent later in the program. This is done with 8 switch
* buttons and LEDs and a jumper wire that is
* used to cut off communication and to test functionality. Lastly the program uses serial
* communication to send a string
* and display it on the LCD without any extra characters.
*
* Hardware:
* Arduino ATmega2560 micro controller
* Communications (USART):
* TX0 = PINE1
* RX0 = PINE0
* Outputs:
* LED0-7   =   PINA0-7
* Inputs:
* SW0-6    =   PINL0-6
* LCD connections:
* RS = PIND0
* RW = PIND1
* EN = PIND2
* D0-D7 = PINL0-7
* A     = 5V
* K     = GND
* VSS   = GND
* VDD   = 5V
* VE    = 10kohm potentiometer
*****************************************************************************/



//////////////////////////////////////////////////////////////////////////////
//checkoff 1
//Objective: Create modules for LCD functions. Verify and display test message.
//////////////////////////////////////////////////////////////////////////////
#include <avr/io.h>
#include <util/delay.h>
//#include "Debugger.h"
#include "LCD.h"

int main(void)
{
	char T_mess[] = {"Test"};
	char B_mess[] = {"Test"};

	Init_PORTs();
	Init_LCD();

	LCD_write(INSTR_WR, 0x01);
	_delay_ms(20);		//can remove if use Busy Flag check
```

```
	LCD_write(INSTR_WR, 0x02);
	_delay_ms(20);  //can remove if use Busy Flag check

	LCD_write(INSTR_WR, 0x86);//sets to top middle of LCD
	Print_string (T_mess);


	LCD_write(INSTR_WR, 0xC0);
	_delay_us(50);       //can remove if use Busy Flag check


	while(1)
	{

		LCD_write(INSTR_WR, 0xC6);// Sets to bottom middle of LCD
		_delay_us(50);       //can remove if use Busy Flag check
		Print_string (B_mess);
		_delay_ms(50);
	}
		return(0);
	}

////////////////////////////////////////////////////////////////////////////////
//checkoff 2
//Objective: Create USART modules. Demonstrate USART communication with jumper wire,
//switch buttons, and LEDs
////////////////////////////////////////////////////////////////////////////////

	#include <avr/io.h>
	#include <avr/interrupt.h>
	#include "USART0.h"

	int main(void)
	{
		init_UART0();
		init_PORTs();
	 sei();

	    while (1) {
			uint8_t switchValue = read_switches(); //send switch value to
readswitch function
			UART_out(switchValue);
			   PORTA=rx_char; // Display the received character on LEDs
		}

	}

void init_PORTs (void)
{
	//inputs
	DDRL = 0x00;  // PORTL all inputs
	PORTL = 0x00; // Set all outputs to 0 initially

	//outputs
	DDRA = 0xFF;//Set PORTA as LED output
	PORTA = 0x00; //Turn LEDS off at initialization
}
```

```
////////////////////////////////////////////////////////////////////////////////
//checkoff 3
//Objective: Demonstrate use of both LCD and USART module. Display a string on the LCD.
////////////////////////////////////////////////////////////////////////////////

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include "main.h"

volatile uint8_t LCD_update;


int main(void) {
	init_PORTs();
	init_LCD();
 LCD_write(INSTR_WR, 0x01);  // Clear display
 _delay_ms(2);               // Delay for LCD to complete the operation
 LCD_write(INSTR_WR, 0x02);  // Set cursor to home
 _delay_ms(2);               // Delay for LCD to complete the operation
	init_UART0(); //initialize USART
		sei(); //enable global interrupts

	while (1) {
		if(LCD_update){
		LCD_update=0;
		LCD_write(INSTR_WR, 0x86); // Set to top middle of LCD
		Print_string(rx_buffer); //write text in array to LCD
		 _delay_ms(50);
		}
		}
	}

/*
 * LCD.c
 *
 * Created: 11/14/2023 7:17:04 PM
 * Author : Caiden Moreno
 */

////////////////////////////////////////////////////////////////////////////////
//checkoff 1
//Objective: Create modules for LCD functions. Verify and display test message.
////////////////////////////////////////////////////////////////////////////////

#include <avr/io.h>
#include "LCD.h"

void Init_PORTs (void)
{
	DDRL = 0xFF;  // PORTL all outputs
	PORTL = 0x00; // Set all outputs to 0 initially
	DDRD = 0x07;  // Set D.0 through A.2 to outputs for LCD control
}


void Init_LCD (void)
{
```

```
	_delay_ms(35);				// wait for more than 30mS after VDD
rises to 4.5V
	LCD_write(INSTR_WR,0x38);	// function set 8bits, 2line, display off
	_delay_us(50);				// wait for more than 39microS
	LCD_write(INSTR_WR,0x0C);	// display on, cursor off, blink off
	_delay_us(50);				// wait for more than 39microS
	LCD_write(INSTR_WR,0x01);	// display clear
	_delay_ms(2);				// wait for more than 1.53mS
	LCD_write(INSTR_WR,0x06);	// entry mode set, increment mode
}


void LCD_write (unsigned char RS, unsigned char data)
{
	if(RS==DATA_WR) PORTD = 0b00000001;	// write data: E = 0 R/!W=0, (write)RS =
1 ,
	else		PORTD = 0b00000000;	// Write instruction: RS = 0 E = 0,
R/!W=0 (write)


	PORTD = PORTD | 0x04;	// Take E HIGH (logic 1)
	PORTL = data;
	_delay_us(50);			// needs to be at least 30uS or no display - use
50
	PORTD = PORTD & 0x01;	// Take E LOW (logic 0)
	_delay_us(50);			// Delay REQUIRED
}

void Print_string(char *str_ptr)
{
	PORTD = 0b00000001;	// write data: RS = 1 E = 0, R/!W=0 (write)

	while(*str_ptr != '\0')
	{
		PORTD = PORTD | 0x04;	// Take E HIGH (logic 1)
		PORTL = *str_ptr++;
		_delay_us(50);			// needs to be at least 30uS or no
display - use 50
		PORTD = PORTD & 0x01;	// Take E LOW (logic 0)
		_delay_us(50);			// Delay REQUIRED
	}
}

////////////////////////////////////////////////////////////////////////////////
//checkoff 3
//Objective: Demonstrate use of both LCD and USART module. Display a string on the LCD.
////////////////////////////////////////////////////////////////////////////////

#include <avr/io.h>
#include "LCD.h"

void init_PORTs (void)
{
	DDRL = 0xFF;	// PORTL all outputs
	PORTL = 0x00; // Set all outputs to 0 initially
	DDRD = 0x07;	// Set D.0 through A.2 to outputs for LCD control
}
```

```
void init_LCD (void)
{
	_delay_ms(35);				// wait for more than 30mS after VDD
rises to 4.5V
	LCD_write(INSTR_WR,0x38);	// function set 8bits, 2line, display off
	_delay_us(50);				// wait for more than 39microS
	LCD_write(INSTR_WR,0x0C);	// display on, cursor off, blink off
	_delay_us(50);				// wait for more than 39microS
	LCD_write(INSTR_WR,0x01);	// display clear
	_delay_ms(2);				// wait for more than 1.53mS
	LCD_write(INSTR_WR,0x06);	// entry mode set, increment mode
}


void LCD_write (unsigned char RS, unsigned char data)
{
	if(RS==DATA_WR) PORTD = 0b00000001;	// write data: E = 0 R/!W=0, (write)RS =
1 ,
	else		PORTD = 0b00000000;	// Write instruction: RS = 0 E = 0,
R/!W=0 (write)


	PORTD = PORTD | 0x04;	// Take E HIGH (logic 1)
	PORTL = data;
	_delay_us(50);			// needs to be at least 30uS or no display - use
50
	PORTD = PORTD & 0x01;	// Take E LOW (logic 0)
	_delay_us(50);			// Delay REQUIRED
}

void Print_string(char *str_ptr)
{
	PORTD = 0b00000001;  // write data: RS = 1 E = 0, R/!W=0 (write)

	while(*str_ptr != '\0')
	{
		PORTD = PORTD | 0x04;	// Take E HIGH (logic 1)
		PORTL = *str_ptr++;
		_delay_us(50);			// needs to be at least 30uS or no
display - use 50
		PORTD = PORTD & 0x01;	// Take E LOW (logic 0)
		_delay_us(50);			// Delay REQUIRED
	}
}
```

```c
/*
 * LCD.h
 *
 * Created: 11/14/2023 7:17:21 PM
 *  Author: Caiden Moreno
 */
////////////////////////////////////////////////////////////////////////////////
//checkoff 1
//Objective: Create modules for LCD functions. Verify and display test message.
////////////////////////////////////////////////////////////////////////////////
#ifndef LCD_H_
#define LCD_H_

//defines
#define F_CPU 16000000UL //clock frequency (16MHz)

//include files
#include <avr/io.h>
#include <util/delay.h>

#define INSTR_WR     0
#define DATA_WR             1

//Define function Prototypes
void Init_PORTs (void);
void Init_LCD (void);
void LCD_write (unsigned char RS, unsigned char data);
void Print_string(char *data_ptr);

#endif  // LCD_H_

////////////////////////////////////////////////////////////////////////////////
//checkoff 3
//Objective: Demonstrate use of both LCD and USART module. Display a string on the LCD.
////////////////////////////////////////////////////////////////////////////////

#ifndef LCD_H_
#define LCD_H_

//defines
#define F_CPU 16000000UL //clock frequency (16MHz)

//include files
#include <avr/io.h>
#include <util/delay.h>

#define INSTR_WR     0
#define DATA_WR             1

//Define function Prototypes
void init_PORTs (void);
void init_LCD (void);
void LCD_write (unsigned char RS, unsigned char data);
void Print_string(char *data_ptr);

#endif  // LCD_H_
```

```c
/*
 * USART0.c
 *
 * Created: 11/16/2023 2:07:22 PM
 * Author : Caiden Moreno
 */
////////////////////////////////////////////////////////////////////////////////
//checkoff 2
//Objective: Create USART modules. Demonstrate USART communication with jumper wire,
//switch buttons, and LEDs
////////////////////////////////////////////////////////////////////////////////
#include <avr/io.h>
#include "USART0.h"

void init_UART0(void)
{
	UCSR0A=0x00; //clear USCR0A
	// Enable receiver and transmitter, enable RX complete interrupt
	UCSR0B = (1 << RXCIE0) | (1 << RXEN0) | (1 << TXEN0);
	UCSR0C=(1<<UCSZ01) | (1<<UCSZ00); //8 bit data
	uint16_t myubr = (F_CPU)/(16UL*BAUD)-1; //9600 baud rate no parity 1 stop bit
	UBRR0L= myubr; //load UBBR low
	UBRR0H=(myubr>>8) & 0x0F; //load UBBR high
}

void UART_out (uint8_t ch) //transmit byte of data
{
  while((UCSR0A & (1<<UDRE0))==0)
  {
	  //wait for UDRE0 flag to indicate UDR0 ready for new data to start new
transmission
  }
  UDR0=ch;//store charcter in UDR0
}

ISR(USART0_RX_vect)
{
	rx_char=UDR0;  // Read the received character from UART

}

uint8_t read_switches(void) {
	return PINL;
}


////////////////////////////////////////////////////////////////////////////////
//checkoff 3
//Objective: Demonstrate use of both LCD and USART module. Display a string on the LCD.
////////////////////////////////////////////////////////////////////////////////

#include <avr/io.h>
#include "USART0.h"

void init_UART0(void)
{
	UCSR0A=0x00; //clear USCR0A
	// Enable transmit, receive, and receive interrupt
```

```
	UCSR0B = (1 << RXCIE0) | (1 << RXEN0) | (1 << TXEN0);
	UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // 8-bit data

	uint16_t myubr = (F_CPU)/(16UL*BAUD)-1; //9600 baud rate no parity 1 stop bit
	UBRR0L= myubr; //load UBBR low
	UBRR0H=(myubr>>8) & 0x0F; //load UBBR high
}

uint8_t UART_in (void) //transmit byte of data
{
	while((UCSR0A & (1<<RXC0))==0)
	{
		//wait for data to arrive
	}
	return (UDR0);//store charcter in UDR0
}

// Interrupt Service Routine (ISR) for USART0 receive vector
ISR(USART0_RX_vect) {
	   rx_char = UDR0;          // Read the received character from UART
	      if(rx_char!=0x0D){      // Check if the received character is not a
carriage return (0x0D)
	      // Store the received character in the buffer and increment the array index
	      rx_buffer[array_index]=rx_char;
	      array_index++;
	      }
	      else{
			rx_buffer[array_index] = '\0';  // If the received character is a
carriage return, terminate the buffer with null character
			array_index=0; //reset array index
			LCD_update=1; //set flag to one to indicate data is ready
	      }

}

/*
 * USART0.h
 *
 * Created: 11/16/2023 2:07:53 PM
 *  Author: Caiden Moreno
 */
////////////////////////////////////////////////////////////////////////////////
//checkoff 2
//Objective: Create USART modules. Demonstrate USART communication with jumper wire,
//switch buttons, and LEDs
////////////////////////////////////////////////////////////////////////////////
#ifndef USART0_H_
#define USART0_H_

//defines
#define F_CPU 16000000UL //clock frequency (16MHz)
//#define F_CPU 8000000UL //clock frequency (8MHz)
#define BAUD 9600
//include files
#include <avr/io.h>
#include <avr/interrupt.h>

//Define function Prototypes
```

```
void init_UART0(void);
void UART_out (uint8_t ch);
void init_PORTs(void);
uint8_t read_switches(void);

//global variables
volatile uint8_t rx_char;
volatile uint8_t value;

#endif  //USART0_H_


////////////////////////////////////////////////////////////////////////////////
//checkoff 3
//Objective: Demonstrate use of both LCD and USART module. Display a string on the LCD.
////////////////////////////////////////////////////////////////////////////////

#ifndef USART0_H_
#define USART0_H_

//defines
#define F_CPU 16000000UL //clock frequency (16MHz)
//#define F_CPU 8000000UL //clock frequency (8MHz)
#define BAUD 9600
//include files
#include <avr/io.h>
#include <avr/interrupt.h>

//Define function Prototypes
void init_UART0(void);
uint8_t UART_in (void);


//global variables
volatile uint8_t rx_char;
volatile uint8_t array_index;
volatile uint8_t LCD_update;
volatile char rx_buffer[25];

#endif  //USART0_H_
```

```
*
 * main.h
 *
 * Created: 11/19/2023 5:36:47 PM
 *  Author: Caiden Moreno
 */
////////////////////////////////////////////////////////////////////////////////
//checkoff 3
//Objective: Demonstrate use of both LCD and USART module. Display a string on the LCD.
////////////////////////////////////////////////////////////////////////////////

#ifndef MAIN_H_
#define MAIN_H_

#include "LCD.h"
#include "USART0.h"


// Global variables
 volatile uint8_t array_index=0;
 volatile uint8_t LCD_update;
 volatile char rx_buffer[25];
 volatile uint8_t rx_char;

#endif // MAIN_H_
```