Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

CENG 347 Lab 4

Introduction:

In this lab, we will interface an 8x8 LED matrix with an ATmega2560 microcontroller to create a scrolling text display. Using the MAX7219 driver, we'll first display numbers and uppercase letters. Next, we'll add serial communication via USART0, allowing custom strings from the terminal to be displayed. Finally, we'll connect two matrices together to display two characters at once, creating a continuous scrolling effect. This project demonstrates hardware interfacing, serial communication, and synchronized matrix control.

Procedure:

The lab is divided into three key parts:

Part 1: Interface with Matrix

- We were to write to each row of the matrix using a lookup table for digits 0–9 and letters A–Z. Then cycle through the lookup table with short delays to verify that every character displays correctly. We each wrote our own code for this section.

Part 2: Terminal Serial Communication

- We were to set up USART0 with 9600 BAUD, 8-bit data, 1 stop bit, and no parity. Using the serial monitor we sent a string to the microcontroller, which then echoed the input and displayed the string on the LED matrix. This part demonstrates how to integrate serial communication with the LED display. We each wrote our own code for this section.

Part 3: Serial Matrix Communication

- We were to connect the output of one matrix to the input of the other matrix so that two characters are shown simultaneously to create continuous scrolling by synchronizing both LED matrices. We worked on this section of the lab together since we had to share hardware to do this.

Application:

Part 1: Interface with Matrix

Code: (Alexis) This code was written by me to be able to be run for part 1 and 2 so i could reuse functions. So this code is for parts 1 and 2.

```
#define LAB_PART 2 //switch to a 1 or 2 to run part 1 or 2 code

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

#define BAUD 9600
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
#define UBRR_VALUE ((F_CPU / (16UL * BAUD)) - 1)
#define DIN   PA0  //din
#define CLK   PA1  //clk
#define CS    PA2  //cs

//given lookup table
unsigned char charTable[38][8] =
{
   //0
   {0x3C,0x42,0x42,0x42,0x42,0x42,0x42,0x3C},
   //1
   {0x10,0x18,0x14,0x10,0x10,0x10,0x10,0x10},
   //2
   {0x7E,0x02,0x02,0x7E,0x40,0x40,0x40,0x7E},
   //3
   {0x3E,0x02,0x02,0x3E,0x02,0x02,0x3E,0x00},
   //4
   {0x08,0x18,0x28,0x48,0xFE,0x08,0x08,0x08},
   //5
   {0x3C,0x20,0x20,0x3C,0x04,0x04,0x3C,0x00},
   //6
   {0x3C,0x20,0x20,0x3C,0x24,0x24,0x3C,0x00},
   //7
   {0x3E,0x22,0x04,0x08,0x08,0x08,0x08,0x08},
   //8
   {0x00,0x3E,0x22,0x22,0x3E,0x22,0x22,0x3E},
   //9
   {0x3E,0x22,0x22,0x3E,0x02,0x02,0x02,0x3E},
   //A
   {0x08,0x14,0x22,0x3E,0x22,0x22,0x22,0x22},
   //B
   {0x3C,0x22,0x22,0x3E,0x22,0x22,0x3C,0x00},
   //C
   {0x3C,0x40,0x40,0x40,0x40,0x40,0x3C,0x00},
   //D
   {0x7C,0x42,0x42,0x42,0x42,0x42,0x7C,0x00},
   //E
   {0x7C,0x40,0x40,0x7C,0x40,0x40,0x40,0x7C},
   //F
   {0x7C,0x40,0x40,0x7C,0x40,0x40,0x40,0x40},
   //G
   {0x3C,0x40,0x40,0x40,0x40,0x44,0x44,0x3C},
   //H
```

```
    {0x44,0x44,0x44,0x7C,0x44,0x44,0x44,0x44},
    //I
    {0x7C,0x10,0x10,0x10,0x10,0x10,0x10,0x7C},
    //J
    {0x3C,0x08,0x08,0x08,0x08,0x08,0x48,0x30},
    //K
    {0x00,0x24,0x28,0x30,0x20,0x30,0x28,0x24},
    //L
    {0x40,0x40,0x40,0x40,0x40,0x40,0x40,0x7C},
    //M
    {0x81,0xC3,0xA5,0x99,0x81,0x81,0x81,0x81},
    //N
    {0x00,0x42,0x62,0x52,0x4A,0x46,0x42,0x00},
    //O
    {0x3C,0x42,0x42,0x42,0x42,0x42,0x42,0x3C},
    //P
    {0x3C,0x22,0x22,0x22,0x3C,0x20,0x20,0x20},
    //Q
    {0x1C,0x22,0x22,0x22,0x22,0x26,0x22,0x1D},
    //R
    {0x3C,0x22,0x22,0x22,0x3C,0x24,0x22,0x21},
    //S
    {0x00,0x1E,0x20,0x20,0x3E,0x02,0x02,0x3C},
    //T
    {0x00,0x3E,0x08,0x08,0x08,0x08,0x08,0x08},
    //U
    {0x42,0x42,0x42,0x42,0x42,0x42,0x22,0x1C},
    //V
    {0x42,0x42,0x42,0x42,0x42,0x42,0x24,0x18},
    //W
    {0x00,0x49,0x49,0x49,0x49,0x2A,0x1C,0x00},
    //X
    {0x00,0x41,0x22,0x14,0x08,0x14,0x22,0x41},
    //Y
    {0x41,0x22,0x14,0x08,0x08,0x08,0x08,0x08},
    //Z
    {0x00,0x7F,0x02,0x04,0x08,0x10,0x20,0x7F}
};

void initUSART(unsigned int ubrr)
{
    UBRR0H = (unsigned char)(ubrr >> 8);
    UBRR0L = (unsigned char)ubrr;
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
   UCSR0B = (1 << RXEN0) | (1 << TXEN0);
   UCSR0C = (1 << UCSZ01) | (1 << UCSZ00);
}

void TX(unsigned char data)
{
   while (!(UCSR0A & (1 << UDRE0)));
   UDR0 = data;
}

unsigned char RX(void)
{
   while (!(UCSR0A & (1 << RXC0)));
   return UDR0;
}

void write(unsigned char data)
{
   for (int i = 0; i < 8; i++)
   {
       PORTA &= ~(1 << CLK);
       if (data & 0x80)
           PORTA |= (1 << DIN);
       else
           PORTA &= ~(1 << DIN);
       PORTA |= (1 << CLK); //load bit
       data <<= 1;
   }
}

void writeAddr(unsigned char addr, unsigned char data)
{
   PORTA &= ~(1 << CS);  //transmit
   write(addr);
   write(data);
   PORTA |= (1 << CS);   //latch
}

void initLED(void)
{
   //set outputs
   DDRA |= (1 << DIN) | (1 << CS) | (1 << CLK);
   writeAddr(0x09, 0x00);
```

```
    _delay_ms(10);
    writeAddr(0x0A, 0x03); //brightness = 3
    _delay_ms(10);
    writeAddr(0x0B, 0x07); //display all rows
    _delay_ms(10);
    writeAddr(0x0C, 0x01); //exit shutdown
    _delay_ms(10);
    writeAddr(0x0F, 0x00);
    _delay_ms(10);
}

void clearDisplay(void)
{
    for (int row = 1; row <= 8; row++)
    {
        writeAddr(row, 0x00);
    }
}

void displayChar(char value)
{
    //lower to upper
    if (value >= 'a' && value <= 'z')
    {
        value -= 32;
    }

    uint8_t index;
    if (value >= '0' && value <= '9')
    {
        index = value - '0';
    }
    else if (value >= 'A' && value <= 'Z')
    {
        index = value - 'A' + 10;
    }
    else
    {
        clearDisplay();
        return;
    }

    //display value
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
    for (int i = 0; i < 8; i++)
    {
        writeAddr(i + 1, charTable[index][i]);
    }
}

void part1(void)
{
    while (true)
    {
        for (uint8_t i = 0; i < 36; i++)
        {
            //display the table
            for (int row = 0; row < 8; row++)
            {
                writeAddr(row + 1, charTable[i][row]);
            }
            _delay_ms(500);
            clearDisplay();
            _delay_ms(100);
        }
    }
}

void part2(void)
{
    char receivedString[81] = {0}; //80 char buffer
    uint8_t index = 0;

    while (true)
    {
        char receivedChar = RX();
        TX(receivedChar);  //echo

        if (receivedChar == '\n' || receivedChar == '\r' || index >= 79)
        {
            receivedString[index] = '\0';
            //display the string
            for (int i = 0; receivedString[i] != '\0'; i++)
            {
                displayChar(receivedString[i]);
                _delay_ms(500);
                clearDisplay();
```

```
                _delay_ms(100);
            }
            index = 0; //reset buffer
        }
        else
        {
            receivedString[index++] = receivedChar;
        }
    }
}

int main(void)
{
    initLED();
    //if else for choosing what part of the lab to run
    #if LAB_PART == 2
        initUSART(UBRR_VALUE);
        _delay_ms(1000);
        clearDisplay();
        _delay_ms(500);
        part2();
    #else
        part1();
    #endif

    return 0;
}
```

Code: (Zack)

```
#include <avr/io.h>
#include <util/delay.h>
//#include <Arduino.h>

#define USART_BAUDRATE 57600
#define BAUD_PRESCALE F_CPU / (USART_BAUDRATE * 16UL) -1

// put function declarations here:

const int CLK = PB4; //Pin 10
const int DIN = PB5; //Pin 11
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
const int CS  = PB6; //Pin 12



#define F_CPU 16000000UL  // 16 MHz clock speed
#define BAUD 9600
#define MYUBRR F_CPU/16/BAUD-1
#define MAX_STR_LEN 80
void USART0_transmit(unsigned char data)
{
    while (!(UCSR0A & (1 << UDRE0))); // Wait for empty transmit buffer
    UDR0 = data;
}

void USART0_init(unsigned int ubrr)
{
    UBRR0H = (unsigned char)(ubrr >> 8);
    UBRR0L = (unsigned char)ubrr;
    UCSR0B = (1 << RXEN0) | (1 << TXEN0);  // Enable receiver and transmitter
    UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // 8-bit data, 1 stop bit, no parity
}

// Character lookup table (example for 0-9 and A-Z)
const unsigned char charTable[36][8] = {
  {0x3E, 0x51, 0x49, 0x45, 0x3E, 0x00, 0x00, 0x00}, // 0
  {0x00, 0x42, 0x7F, 0x40, 0x00, 0x00, 0x00, 0x00}, // 1
  {0x62, 0x51, 0x49, 0x49, 0x46, 0x00, 0x00, 0x00}, // 2
  {0x22, 0x41, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 3
  {0x18, 0x14, 0x12, 0x7F, 0x10, 0x00, 0x00, 0x00}, // 4
  {0x27, 0x45, 0x45, 0x45, 0x39, 0x00, 0x00, 0x00}, // 5
  {0x3C, 0x4A, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // 6
  {0x01, 0x71, 0x09, 0x05, 0x03, 0x00, 0x00, 0x00}, // 7
  {0x36, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 8
  {0x46, 0x49, 0x49, 0x29, 0x1E, 0x00, 0x00, 0x00}, // 9
  {0x7E, 0x09, 0x09, 0x09, 0x7E, 0x00, 0x00, 0x00}, // A
  {0x7F, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // B
  {0x3E, 0x41, 0x41, 0x41, 0x22, 0x00, 0x00, 0x00}, // C
  {0x7F, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // D
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
  {0x7F, 0x49, 0x49, 0x49, 0x41, 0x00, 0x00, 0x00}, // E
  {0x7F, 0x09, 0x09, 0x09, 0x01, 0x00, 0x00, 0x00}, // F
  {0x3E, 0x41, 0x49, 0x49, 0x7A, 0x00, 0x00, 0x00}, // G
  {0x7F, 0x08, 0x08, 0x08, 0x7F, 0x00, 0x00, 0x00}, // H
  {0x00, 0x41, 0x7F, 0x41, 0x00, 0x00, 0x00, 0x00}, // I
  {0x20, 0x40, 0x41, 0x3F, 0x01, 0x00, 0x00, 0x00}, // J
  {0x7F, 0x08, 0x14, 0x22, 0x41, 0x00, 0x00, 0x00}, // K
  {0x7F, 0x40, 0x40, 0x40, 0x40, 0x00, 0x00, 0x00}, // L
  {0x7F, 0x02, 0x04, 0x02, 0x7F, 0x00, 0x00, 0x00}, // M
  {0x7F, 0x02, 0x04, 0x08, 0x7F, 0x00, 0x00, 0x00}, // N
  {0x3E, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // O
  {0x7F, 0x09, 0x09, 0x09, 0x06, 0x00, 0x00, 0x00}, // P
  {0x3E, 0x41, 0x51, 0x21, 0x5E, 0x00, 0x00, 0x00}, // Q
  {0x7F, 0x09, 0x19, 0x29, 0x46, 0x00, 0x00, 0x00}, // R
  {0x46, 0x49, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // S
  {0x01, 0x01, 0x7F, 0x01, 0x01, 0x00, 0x00, 0x00}, // T
  {0x3F, 0x40, 0x40, 0x40, 0x3F, 0x00, 0x00, 0x00}, // U
  {0x0F, 0x30, 0x40, 0x30, 0x0F, 0x00, 0x00, 0x00}, // V
  {0x7F, 0x20, 0x18, 0x20, 0x7F, 0x00, 0x00, 0x00}, // W
  {0x63, 0x14, 0x08, 0x14, 0x63, 0x00, 0x00, 0x00}, // X
  {0x07, 0x08, 0x70, 0x08, 0x07, 0x00, 0x00, 0x00}, // Y
  {0x61, 0x51, 0x49, 0x45, 0x43, 0x00, 0x00, 0x00}  // Z
};



// Function to send a byte to MAX7219
void WriteByte(unsigned char DATA)
{
  for (int i = 0; i < 8; i++)
  {
      PORTB &= ~(1 << CLK);  // Clear clock bit
      if (DATA & 0x80)        // Check MSB
      {
        PORTB |= (1 << DIN);  // Set DIN high
      }
      else
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
      {
        PORTB &= ~(1 << DIN); // Set DIN low
      }
      DATA <<= 1;             // Shift left
      PORTB |= (1 << CLK);    // Set clock to load bit
  }
}


// Function to send address + data to MAX7219
void WriteData(unsigned char addr, unsigned char data)
{
  USART0_transmit(data);
  //Serial.println(addr);
  //USART0_Transmit(data - '0');
  PORTB &= ~(1 << CS);  // Clear CS (start communication)
  WriteByte(addr);      // Send address
  WriteByte(data);      // Send data
  PORTB |= (1 << CS);   // Set CS (latch data)
}


// MAX7219 initialization
void InitMatrix() {
  WriteData(0x09, 0x00);  // Decode mode: No decode
  WriteData(0x0A, 0x03);  // Intensity: Medium
  WriteData(0x0B, 0x07);  // Scan limit: Display all rows
  WriteData(0x0C, 0x01);  // Shutdown: Normal operation
  WriteData(0x0F, 0x00);  // Display test: Off
}

// Function to display a character on LED matrix
void Display_Character(unsigned char index)
{
  //USART0_Transmit(index);
  for (int i = 0; i < 8; i++) {
      WriteData(i + 1, charTable[index][i]);
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
      //USART0_Transmit(char(charTable[index][i]));
  }
}


int main()
{
    DDRB |= (1 << DIN) | (1 << CS) | (1 << CLK);
        InitMatrix();  // Initialize MAX7219 LED matrix

    while (1)
    {
        for(int i =0; i < 32; i++)
        {
          Display_Character(i);
          _delay_ms(500);
        }
    }

    return 0;
}
```

Part 2: Terminal Serial Communication

Code(Zack): None Interrupt Base

```
#include <avr/io.h>
#include <util/delay.h>
#include <string.h>

#define F_CPU 16000000UL  // 16 MHz clock speed
#define BAUD 9600
#define MYUBRR F_CPU/16/BAUD-1
#define MAX_STR_LEN 80

// put function declarations here:

const int CLK = PB4; //Pin 10
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
const int DIN = PB5; //Pin 11
const int CS  = PB6; //Pin 12

// Character lookup table (example for 0-9 and A-Z)
const unsigned char charTable[36][8] = {
  {0x3E, 0x51, 0x49, 0x45, 0x3E, 0x00, 0x00, 0x00}, // 0
  {0x00, 0x42, 0x7F, 0x40, 0x00, 0x00, 0x00, 0x00}, // 1
  {0x62, 0x51, 0x49, 0x49, 0x46, 0x00, 0x00, 0x00}, // 2
  {0x22, 0x41, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 3
  {0x18, 0x14, 0x12, 0x7F, 0x10, 0x00, 0x00, 0x00}, // 4
  {0x27, 0x45, 0x45, 0x45, 0x39, 0x00, 0x00, 0x00}, // 5
  {0x3C, 0x4A, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // 6
  {0x01, 0x71, 0x09, 0x05, 0x03, 0x00, 0x00, 0x00}, // 7
  {0x36, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 8
  {0x46, 0x49, 0x49, 0x29, 0x1E, 0x00, 0x00, 0x00}, // 9
  {0x7E, 0x09, 0x09, 0x09, 0x7E, 0x00, 0x00, 0x00}, // A
  {0x7F, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // B
  {0x3E, 0x41, 0x41, 0x41, 0x22, 0x00, 0x00, 0x00}, // C
  {0x7F, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // D
  {0x7F, 0x49, 0x49, 0x49, 0x41, 0x00, 0x00, 0x00}, // E
  {0x7F, 0x09, 0x09, 0x09, 0x01, 0x00, 0x00, 0x00}, // F
  {0x3E, 0x41, 0x49, 0x49, 0x7A, 0x00, 0x00, 0x00}, // G
  {0x7F, 0x08, 0x08, 0x08, 0x7F, 0x00, 0x00, 0x00}, // H
  {0x00, 0x41, 0x7F, 0x41, 0x00, 0x00, 0x00, 0x00}, // I
  {0x20, 0x40, 0x41, 0x3F, 0x01, 0x00, 0x00, 0x00}, // J
  {0x7F, 0x08, 0x14, 0x22, 0x41, 0x00, 0x00, 0x00}, // K
  {0x7F, 0x40, 0x40, 0x40, 0x40, 0x00, 0x00, 0x00}, // L
  {0x7F, 0x02, 0x04, 0x02, 0x7F, 0x00, 0x00, 0x00}, // M
  {0x7F, 0x02, 0x04, 0x08, 0x7F, 0x00, 0x00, 0x00}, // N
  {0x3E, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // O
  {0x7F, 0x09, 0x09, 0x09, 0x06, 0x00, 0x00, 0x00}, // P
  {0x3E, 0x41, 0x51, 0x21, 0x5E, 0x00, 0x00, 0x00}, // Q
  {0x7F, 0x09, 0x19, 0x29, 0x46, 0x00, 0x00, 0x00}, // R
  {0x46, 0x49, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // S
  {0x01, 0x01, 0x7F, 0x01, 0x01, 0x00, 0x00, 0x00}, // T
  {0x3F, 0x40, 0x40, 0x40, 0x3F, 0x00, 0x00, 0x00}, // U
  {0x0F, 0x30, 0x40, 0x30, 0x0F, 0x00, 0x00, 0x00}, // V
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
  {0x7F, 0x20, 0x18, 0x20, 0x7F, 0x00, 0x00, 0x00}, // W
  {0x63, 0x14, 0x08, 0x14, 0x63, 0x00, 0x00, 0x00}, // X
  {0x07, 0x08, 0x70, 0x08, 0x07, 0x00, 0x00, 0x00}, // Y
  {0x61, 0x51, 0x49, 0x45, 0x43, 0x00, 0x00, 0x00}  // Z
};

void USART0_transmit(unsigned char data)
{
    while (!(UCSR0A & (1 << UDRE0))); // Wait for empty transmit buffer
    UDR0 = data;
}

// Function to send a byte to MAX7219
void WriteByte(unsigned char DATA)
{
  for (int i = 0; i < 8; i++)
  {
      PORTB &= ~(1 << CLK);  // Clear clock bit
      if (DATA & 0x80)        // Check MSB
      {
        PORTB |= (1 << DIN);  // Set DIN high
      }
      else
      {
        PORTB &= ~(1 << DIN); // Set DIN low
      }
      DATA <<= 1;             // Shift left
      PORTB |= (1 << CLK);    // Set clock to load bit
  }
}


// Function to send address + data to MAX7219
void WriteData(unsigned char addr, unsigned char data)
{
  //Serial.println(addr);
  //USART0_transmit(data);
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
  PORTB &= ~(1 << CS);  // Clear CS (start communication)
  WriteByte(addr);      // Send address
  WriteByte(data);      // Send data
  PORTB |= (1 << CS);   // Set CS (latch data)
}


// MAX7219 initialization
void InitMatrix() {
  WriteData(0x09, 0x00);  // Decode mode: No decode
  WriteData(0x0A, 0x03);  // Intensity: Medium
  WriteData(0x0B, 0x07);  // Scan limit: Display all rows
  WriteData(0x0C, 0x01);  // Shutdown: Normal operation
  WriteData(0x0F, 0x00);  // Display test: Off
}

void USART0_init(unsigned int ubrr)
{
    UBRR0H = (unsigned char)(ubrr >> 8);
    UBRR0L = (unsigned char)ubrr;
    UCSR0B = (1 << RXEN0) | (1 << TXEN0);  // Enable receiver and transmitter
    UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // 8-bit data, 1 stop bit, no parity
}

unsigned char USART0_receive(void)
{
    while (!(UCSR0A & (1 << RXC0))); // Wait for data to be received
    return UDR0;
}

void Display_Character(unsigned char index)
{
  //USART0_transmit(index);
  for (int i = 0; i < 8; i++)
  {
      WriteData(i + 1, charTable[index][i]);
      //USART0_transmit(charTable[index][i]);
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
  }
}

int main(void) {
    DDRB |= (1 << DIN) | (1 << CS) | (1 << CLK);
    USART0_init(MYUBRR);
    InitMatrix();
    //USART0_sendString("USART Initialized. Enter text:\r\n");

    char receivedString[MAX_STR_LEN];
    int index = 0;

    while (1) {
        char receivedChar = USART0_receive();
        //USART0_transmit(receivedChar);
         // Echo back

        if (receivedChar == '\n' || receivedChar == '\r' || index >= MAX_STR_LEN
- 1)
        {
            receivedString[index] = '\0'; // Null-terminate the string
            for (uint8_t i = 0; i < index; i++) {

                Display_Character(int(receivedString[i]));
                USART0_transmit(receivedString[i]);
                _delay_ms(500); // Delay between characters
            }
            index = 0; // Reset for next input
        }
        else
        {
            if(receivedChar >= '0' && receivedChar <= '9')
            {
                receivedString[index++] = receivedChar - '0';
            }
            else if (receivedChar >= 'A' && receivedChar <= 'Z')
            {
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
                receivedString[index++] = receivedChar - 55;
            }
            else if (receivedChar >= 'a' && receivedChar <= 'z')
            {
                receivedString[index++] = receivedChar - 87;
            }
            //receivedString[index++] = receivedChar;
        }
    }
}
```

Code(Zack): Interrupt Base

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <string.h>

#define F_CPU 16000000UL  // 16 MHz clock speed
#define BAUD 9600
#define MYUBRR F_CPU/16/BAUD-1
#define MAX_STR_LEN 80

const int CLK = PB4; //Pin 10
const int DIN = PB5; //Pin 11
const int CS  = PB6; //Pin 12

volatile  char received_data;
volatile uint8_t data_ready = 0;

// Character lookup table (example for 0-9 and A-Z)
const unsigned char charTable[36][8] = {
    {0x3E, 0x51, 0x49, 0x45, 0x3E, 0x00, 0x00, 0x00}, // 0
    {0x00, 0x42, 0x7F, 0x40, 0x00, 0x00, 0x00, 0x00}, // 1
    {0x62, 0x51, 0x49, 0x49, 0x46, 0x00, 0x00, 0x00}, // 2
    {0x22, 0x41, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 3
    {0x18, 0x14, 0x12, 0x7F, 0x10, 0x00, 0x00, 0x00}, // 4
    {0x27, 0x45, 0x45, 0x45, 0x39, 0x00, 0x00, 0x00}, // 5
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
    {0x3C, 0x4A, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // 6
    {0x01, 0x71, 0x09, 0x05, 0x03, 0x00, 0x00, 0x00}, // 7
    {0x36, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 8
    {0x46, 0x49, 0x49, 0x29, 0x1E, 0x00, 0x00, 0x00}, // 9
    {0x7E, 0x09, 0x09, 0x09, 0x7E, 0x00, 0x00, 0x00}, // A
    {0x7F, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // B
    {0x3E, 0x41, 0x41, 0x41, 0x22, 0x00, 0x00, 0x00}, // C
    {0x7F, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // D
    {0x7F, 0x49, 0x49, 0x49, 0x41, 0x00, 0x00, 0x00}, // E
    {0x7F, 0x09, 0x09, 0x09, 0x01, 0x00, 0x00, 0x00}, // F
    {0x3E, 0x41, 0x49, 0x49, 0x7A, 0x00, 0x00, 0x00}, // G
    {0x7F, 0x08, 0x08, 0x08, 0x7F, 0x00, 0x00, 0x00}, // H
    {0x00, 0x41, 0x7F, 0x41, 0x00, 0x00, 0x00, 0x00}, // I
    {0x20, 0x40, 0x41, 0x3F, 0x01, 0x00, 0x00, 0x00}, // J
    {0x7F, 0x08, 0x14, 0x22, 0x41, 0x00, 0x00, 0x00}, // K
    {0x7F, 0x40, 0x40, 0x40, 0x40, 0x00, 0x00, 0x00}, // L
    {0x7F, 0x02, 0x04, 0x02, 0x7F, 0x00, 0x00, 0x00}, // M
    {0x7F, 0x02, 0x04, 0x08, 0x7F, 0x00, 0x00, 0x00}, // N
    {0x3E, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // O
    {0x7F, 0x09, 0x09, 0x09, 0x06, 0x00, 0x00, 0x00}, // P
    {0x3E, 0x41, 0x51, 0x21, 0x5E, 0x00, 0x00, 0x00}, // Q
    {0x7F, 0x09, 0x19, 0x29, 0x46, 0x00, 0x00, 0x00}, // R
    {0x46, 0x49, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // S
    {0x01, 0x01, 0x7F, 0x01, 0x01, 0x00, 0x00, 0x00}, // T
    {0x3F, 0x40, 0x40, 0x40, 0x3F, 0x00, 0x00, 0x00}, // U
    {0x0F, 0x30, 0x40, 0x30, 0x0F, 0x00, 0x00, 0x00}, // V
    {0x7F, 0x20, 0x18, 0x20, 0x7F, 0x00, 0x00, 0x00}, // W
    {0x63, 0x14, 0x08, 0x14, 0x63, 0x00, 0x00, 0x00}, // X
    {0x07, 0x08, 0x70, 0x08, 0x07, 0x00, 0x00, 0x00}, // Y
    {0x61, 0x51, 0x49, 0x45, 0x43, 0x00, 0x00, 0x00}  // Z
  };

  // Function to send a byte to MAX7219
void WriteByte(unsigned char DATA)
{
  for (int i = 0; i < 8; i++)
  {
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
      PORTB &= ~(1 << CLK);  // Clear clock bit
      if (DATA & 0x80)       // Check MSB
      {
        PORTB |= (1 << DIN);  // Set DIN high
      }
      else
      {
        PORTB &= ~(1 << DIN); // Set DIN low
      }
      DATA <<= 1;            // Shift left
      PORTB |= (1 << CLK);    // Set clock to load bit
  }
}

// Function to send address + data to MAX7219
void WriteData(unsigned char addr, unsigned char data)
{
  //Serial.println(addr);
  //USART0_transmit(data);
  PORTB &= ~(1 << CS);  // Clear CS (start communication)
  WriteByte(addr);      // Send address
  WriteByte(data);      // Send data
  PORTB |= (1 << CS);   // Set CS (latch data)
}

void Display_Character(unsigned char index)
{
  //USART0_transmit(index);
  for (int i = 0; i < 8; i++)
  {
      WriteData(i + 1, charTable[index][i]);
      //USART0_transmit(charTable[index][i]);
  }
}

void USART0_init(unsigned int ubrr) {
    UBRR0H = (unsigned char)(ubrr >> 8);
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
    UBRR0L = (unsigned char)ubrr;
    UCSR0B = (1 << RXEN0) | (1 << TXEN0) | (1 << RXCIE0); // Enable RX, TX, and
RX interrupt
    UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // 8-bit data
    sei(); // Enable global interrupts
}

void USART0_transmit(unsigned char data){
    while (!(UCSR0A & (1 << UDRE0))); // Wait for empty transmit buffer
    UDR0 = data; // Send data
}

ISR(USART_RX_vect){
    received_data = UDR0; // Read received character
    data_ready = 1; // Set flag
}

void SPI_init() {
    DDRB |= (1 << DIN) | (1 << CS) | (1 << CLK); // Set SPI pins as output
    SPCR = (1 << SPE) | (1 << MSTR) | (1 << SPR0); // Enable SPI, Master mode,
set clock rate
}

void SPI_write(unsigned char data) {
    SPDR = data;               // Load data into SPI data register
    while (!(SPSR & (1 << SPIF))); // Wait until transmission complete
}

void WriteData(unsigned char addr, unsigned char data) {
    PORTB &= ~(1 << CS);  // Select device
    SPI_write(addr);
    SPI_write(data);
    PORTB |= (1 << CS);   // Deselect device
}

void InitMatrix() {
    WriteData(0x09, 0x00);
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
    WriteData(0x0A, 0x03);
    WriteData(0x0B, 0x07);
    WriteData(0x0C, 0x01);
    WriteData(0x0F, 0x00);
}

void check_input(unsigned char receivedData)
{
    char receivedString[MAX_STR_LEN];
    int index = 0;

    if (receivedData == '\n' || receivedData == '\r' || index >= MAX_STR_LEN - 1)
        {
            receivedString[index] = '\0'; // Null-terminate the string
            for (uint8_t i = 0; i < index; i++) {

                Display_Character(receivedString[i]);
                USART0_transmit(receivedString[i]);
                _delay_ms(500); // Delay between characters
            }
            index = 0; // Reset for next input
        }
        else
        {
            if(receivedData >= '0' && receivedData <= '9')
            {
                receivedString[index++] = receivedData - '0';
            }
            else if (receivedData >= 'A' && receivedData <= 'Z')
            {
                receivedString[index++] = receivedData - 55;
            }
            else if (receivedData >= 'a' && receivedData <= 'z')
            {
                receivedString[index++] = receivedData - 87;
            }
        }
```

```
}

int main(void) {
    USART0_init(MYUBRR);
    SPI_init();
    InitMatrix();

    while (1) {
        if (data_ready)
        {
            check_input(received_data); // Example action: Send received data to
matrix display
            data_ready = 0;
        }
    }
}
```

Part 3: Terminal Serial Communication

Code:

```
#include <avr/io.h>
#include <util/delay.h>
#include <string.h>

#define F_CPU 16000000UL  // 16 MHz clock speed
#define BAUD 9600
#define MYUBRR F_CPU/16/BAUD-1
#define MAX_STR_LEN 80

// put function declarations here:

const int CLK = PB4; //Pin 10
const int DIN = PB5; //Pin 11
const int CS  = PB6; //Pin 12

// Character lookup table (example for 0-9 and A-Z)
const unsigned char charTable[36][8] = {
  {0x3E, 0x51, 0x49, 0x45, 0x3E, 0x00, 0x00, 0x00}, // 0
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
  {0x00, 0x42, 0x7F, 0x40, 0x00, 0x00, 0x00, 0x00}, // 1
  {0x62, 0x51, 0x49, 0x49, 0x46, 0x00, 0x00, 0x00}, // 2
  {0x22, 0x41, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 3
  {0x18, 0x14, 0x12, 0x7F, 0x10, 0x00, 0x00, 0x00}, // 4
  {0x27, 0x45, 0x45, 0x45, 0x39, 0x00, 0x00, 0x00}, // 5
  {0x3C, 0x4A, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // 6
  {0x01, 0x71, 0x09, 0x05, 0x03, 0x00, 0x00, 0x00}, // 7
  {0x36, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // 8
  {0x46, 0x49, 0x49, 0x29, 0x1E, 0x00, 0x00, 0x00}, // 9
  {0x7E, 0x09, 0x09, 0x09, 0x7E, 0x00, 0x00, 0x00}, // A
  {0x7F, 0x49, 0x49, 0x49, 0x36, 0x00, 0x00, 0x00}, // B
  {0x3E, 0x41, 0x41, 0x41, 0x22, 0x00, 0x00, 0x00}, // C
  {0x7F, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // D
  {0x7F, 0x49, 0x49, 0x49, 0x41, 0x00, 0x00, 0x00}, // E
  {0x7F, 0x09, 0x09, 0x09, 0x01, 0x00, 0x00, 0x00}, // F
  {0x3E, 0x41, 0x49, 0x49, 0x7A, 0x00, 0x00, 0x00}, // G
  {0x7F, 0x08, 0x08, 0x08, 0x7F, 0x00, 0x00, 0x00}, // H
  {0x00, 0x41, 0x7F, 0x41, 0x00, 0x00, 0x00, 0x00}, // I
  {0x20, 0x40, 0x41, 0x3F, 0x01, 0x00, 0x00, 0x00}, // J
  {0x7F, 0x08, 0x14, 0x22, 0x41, 0x00, 0x00, 0x00}, // K
  {0x7F, 0x40, 0x40, 0x40, 0x40, 0x00, 0x00, 0x00}, // L
  {0x7F, 0x02, 0x04, 0x02, 0x7F, 0x00, 0x00, 0x00}, // M
  {0x7F, 0x02, 0x04, 0x08, 0x7F, 0x00, 0x00, 0x00}, // N
  {0x3E, 0x41, 0x41, 0x41, 0x3E, 0x00, 0x00, 0x00}, // O
  {0x7F, 0x09, 0x09, 0x09, 0x06, 0x00, 0x00, 0x00}, // P
  {0x3E, 0x41, 0x51, 0x21, 0x5E, 0x00, 0x00, 0x00}, // Q
  {0x7F, 0x09, 0x19, 0x29, 0x46, 0x00, 0x00, 0x00}, // R
  {0x46, 0x49, 0x49, 0x49, 0x31, 0x00, 0x00, 0x00}, // S
  {0x01, 0x01, 0x7F, 0x01, 0x01, 0x00, 0x00, 0x00}, // T
  {0x3F, 0x40, 0x40, 0x40, 0x3F, 0x00, 0x00, 0x00}, // U
  {0x0F, 0x30, 0x40, 0x30, 0x0F, 0x00, 0x00, 0x00}, // V
  {0x7F, 0x20, 0x18, 0x20, 0x7F, 0x00, 0x00, 0x00}, // W
  {0x63, 0x14, 0x08, 0x14, 0x63, 0x00, 0x00, 0x00}, // X
  {0x07, 0x08, 0x70, 0x08, 0x07, 0x00, 0x00, 0x00}, // Y
  {0x61, 0x51, 0x49, 0x45, 0x43, 0x00, 0x00, 0x00}  // Z
};
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
void USART0_transmit(unsigned char data)
{
    while (!(UCSR0A & (1 << UDRE0))); // Wait for empty transmit buffer
    UDR0 = data;
}

// Function to send a byte to MAX7219
void WriteByte(unsigned char DATA)
{
  for (int i = 0; i < 8; i++)
  {
      PORTB &= ~(1 << CLK);  // Clear clock bit
      if (DATA & 0x80)        // Check MSB
      {
        PORTB |= (1 << DIN);  // Set DIN high
        //PORTB |= (0x0 << DIN);
      }
      else
      {
        PORTB &= ~(1 << DIN); // Set DIN low
      }
      DATA <<= 1;             // Shift left
      PORTB |= (1 << CLK);    // Set clock to load bit
  }
}


// Function to send address + data to MAX7219
void WriteData(unsigned char addr, unsigned char data1, unsigned char data2)
{
  PORTB &= ~(1 << CS);  // Clear CS (start communication)
  WriteByte(addr);      // Send address
  WriteByte(data1);      // Send data
  WriteByte(addr);      // Send address
  WriteByte(data2);      // Send data
  PORTB |= (1 << CS);   // Set CS (latch data)
}
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
// MAX7219 initialization
void InitMatrix() {
  WriteData(0x09, 0x00, 0x00);  // Decode mode: No decode
  WriteData(0x0A, 0x03, 0x03);  // Intensity: Medium
  WriteData(0x0B, 0x07, 0x07);  // Scan limit: Display all rows
  WriteData(0x0C, 0x01, 0x01);  // Shutdown: Normal operation
  WriteData(0x0F, 0x00, 0x00);  // Display test: Off
}

void USART0_init(unsigned int ubrr)
{
    UBRR0H = (unsigned char)(ubrr >> 8);
    UBRR0L = (unsigned char)ubrr;
    UCSR0B = (1 << RXEN0) | (1 << TXEN0);  // Enable receiver and transmitter
    UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // 8-bit data, 1 stop bit, no parity
}

unsigned char USART0_receive(void)
{
    while (!(UCSR0A & (1 << RXC0))); // Wait for data to be received
    return UDR0;
}

void Display_Character(unsigned char index1, unsigned char index2)
{
  //USART0_transmit(index);
  for (int i = 0; i < 8; i++)
  {
      WriteData(i + 1, charTable[index2][i],charTable[index1][i]);
      //USART0_transmit(charTable[index1][i]);
      //USART0_transmit(charTable[index][i]);
  }
}

int main(void) {
```

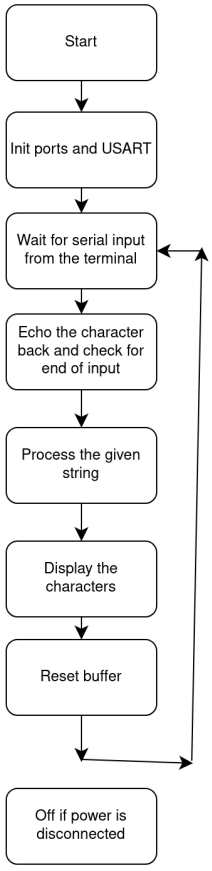
Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
    DDRB |= (1 << DIN) | (1 << CS) | (1 << CLK);
    USART0_init(MYUBRR);
    InitMatrix();
    //USART0_sendString("USART Initialized. Enter text:\r\n");

    char receivedString[MAX_STR_LEN];
    int index = 0;

    while (1) {
        char receivedChar = USART0_receive();
        USART0_transmit(receivedChar);
         // Echo back

        if (receivedChar == '\n' || receivedChar == '\r' || index >= MAX_STR_LEN
- 1)
        {
            receivedString[index] = '\0'; // Null-terminate the string
            for (uint8_t i = 0; i < index; i++)
            {
                //USART0_transmit(receivedString[i]);
                if(receivedString[i] != '\0' && receivedString[i+1] != '\0')
                {
                    Display_Character(receivedString[i], receivedString[i+1]);
                    _delay_ms(500); // Delay between characters
                }

            }
            index = 0; // Reset for next input
        }
        else
        {
            if(receivedChar >= '0' && receivedChar <= '9')
            {
                receivedString[index++] = receivedChar - '0';
            }
            else if (receivedChar >= 'A' && receivedChar <= 'Z')
            {
```

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

```
                receivedString[index++] = receivedChar - 55;
            }
            else if (receivedChar >= 'a' && receivedChar <= 'z')
            {
                receivedString[index++] = receivedChar - 87;
            }
            //receivedString[index++] = receivedChar;
        }
    }
}
```

Flowchart for part 3:



Results:

Part 1: Interface with Matrix

The code ran successfully and as expected. This was verified by the TA visually.

Part 2: Terminal Serial Communication

The code ran successfully and as expected. This was verified by the TA visually.

Alexis Englund, Zackery Holloway
CENG 347 Lab 4
03/26/25

Part 3: Serial Matrix Communication

The code ran successfully and as expected. This was verified by the TA visually.

Conclusion

In this lab, we successfully interfaced and controlled an 8x8 LED matrix using an ATmega2560 microcontroller and the MAX7219 driver. We displayed both predefined and custom text strings, integrating serial communication via USART0 to allow real-time input from the terminal. Finally, by connecting two matrices, we created a scrolling text effect, demonstrating the ability to coordinate multiple displays. This lab strengthened our skills in hardware interfacing, serial communication, and managing synchronized outputs, providing a practical application of embedded systems programming.