ECE 241 Lab 6 Report Spring 2018
Name: Graham Wood Lab Section A

Objective: Using communication hardware to communicate with other devices

Prelab: The code was written and uploaded to the website. The code written and seen below adapts to a LED display and displays a light pattern. The pattern is generated by setting an initial hexadecimal and iterating by another hexadecimal number. If the value gets to big, the program resets the value.

Part 1) In part one, the code was uploaded to the Arduino and observed the pattern on the LED DIsplay. The LED display starts with all the LEDs lit on the first row and each additional row one less column is lit up. The pattern repeats as long as the program is running. The pattern is cause because the initial value is zero and the value is shifted left.

Part 2) In part two the code was modified so that the initial value is 0xc5. This caused the pattern to change and create a staggered pattern that shifted. The pattern follows the initial binary representation of the number, so 1100 0101

Questions:

What would the pattern look like if the initial condition were 0x00AA.

Initial value	1	0	1	0	1	0	1	0
Row 1	0	1	0	1	0	1	0	0
Row 2	1	0	1	0	1	0	0	1
Row 3	0	1	0	1	0	0	1	0
Row 4	1	0	1	0	0	1	0	1
Row 5	0	1	0	0	1	0	1	0
Row 6	1	0	0	1	0	1	0	1
Row 7	0	0	1	0	1	0	1	0
Row 8	0	1	0	1	0	1	0	1

Appendix A) Code for Part 1

```
 int DisplayRow; //Used tto manage the position on the row

int DisplayColumn;
unsigned long timer;
int cs = 10;

#include <SPI.h>

// Code for handling LED display.// MAX7219 SPI LED Driver
#define MAX7219_TEST        0x0f00 // Display in Test mode
#define MAX7219_BRIGHTNESS  0x0a00 // Set brightness of display
#define MAX7219_SCAN_LIMIT  0x0b00 // Set Scan limit
```

```
#define MAX7219_DECODE_MODE 0x0900 // Sets chip to accept bit patterns
#define MAX7219_SHUTDOWN    0x0C00 // Code for shutdown chip.

void SPI_16(int data) {
  SPI.beginTransaction(SPISettings( 8000000, MSBFIRST, SPI_MODE0 )); //begins
the spi transaction
  digitalWrite (cs, LOW); //sets the cs to low
  SPI.transfer16(data); //transfers the data
  digitalWrite (cs, HIGH); //set the cs to high again
  SPI.endTransaction(); //ends the transaction
}

void setup() {
  SPI.begin();
  // put your setup code here, to run once:
  // Set up display These need to be sent in this fashion in setup.
  SPI_16(MAX7219_TEST + 0x01); // Turn on all the LEDs.
  delay(100);                  // One time we can use a delay.
  SPI_16(MAX7219_TEST + 0x00); // all LEDS off.
  SPI_16(MAX7219_DECODE_MODE + 0x00); // Disable BCD mode.
  SPI_16(MAX7219_BRIGHTNESS + 0x03);  // Use lower intensity.
  SPI_16(MAX7219_SCAN_LIMIT + 0x0f);  // Scan all digits.
  SPI_16(MAX7219_SHUTDOWN + 0x01);    // Turn on chip.

  DisplayColumn = 0;// Not valid column, but first pass should increment to
1.
  DisplayRow = 0;// This will effect the pattern generated.


  timer = millis(); //sets timer equal to the amount of milliseconds since
the program has began
  pinMode(10, OUTPUT);

}

void loop() {
  if ((millis() - timer) > 500) { //checks if it has been 500 milliseconds
since the last interval
      DisplayColumn++; //increments the collumn
      if (DisplayColumn > 8) { //if the column is above 8 then it will reset
it back to 1
        DisplayColumn = 1;
      }
      if (DisplayRow & 0x0080) { //if bit 7 is high
        DisplayRow = DisplayRow<<1 ; //shift left
      }
      else { //else
        DisplayRow = DisplayRow<<1 ; //shift left
        DisplayRow |= 0x0001; //force bit 1 to be high
      }
      DisplayRow &= 0x00ff;
      SPI_16 ( ( DisplayColumn <<8 ) + DisplayRow ); //call the spi 16
function

    timer += 500;
  }
```

```
}
```

Appendix B) Code for part 2:

```
// Code for handling LED display.// MAX7219 SPI LED Driver
#define MAX7219_TEST        0x0f00 // Display in Test mode
#define MAX7219_BRIGHTNESS  0x0a00 // Set brightness of display
#define MAX7219_SCAN_LIMIT  0x0b00 // Set Scan limit
#define MAX7219_DECODE_MODE 0x0900 // Sets chip to accept bit patterns
#define MAX7219_SHUTDOWN    0x0C00 // Code for shutdown chip.

void SPI_16(int data) {
  SPI.beginTransaction(SPISettings( 8000000, MSBFIRST, SPI_MODE0 )); //begins
the spi transaction
  digitalWrite (cs, LOW); //sets the cs to low
  SPI.transfer16(data); //transfers the data
  digitalWrite (cs, HIGH); //set the cs to high again
  SPI.endTransaction(); //ends the transaction
}

void setup() {
  SPI.begin();
  // put your setup code here, to run once:
  // Set up display These need to be sent in this fashion in setup.
  SPI_16(MAX7219_TEST + 0x01); // Turn on all the LEDs.
  delay(100);                  // One time we can use a delay.
  SPI_16(MAX7219_TEST + 0x00); // all LEDS off.
  SPI_16(MAX7219_DECODE_MODE + 0x00); // Disable BCD mode.
  SPI_16(MAX7219_BRIGHTNESS + 0x03);  // Use lower intensity.
  SPI_16(MAX7219_SCAN_LIMIT + 0x0f);  // Scan all digits.
  SPI_16(MAX7219_SHUTDOWN + 0x01);    // Turn on chip.

  DisplayColumn = 0;// Not valid column, but first pass should increment to
1.
  DisplayRow = 0x00AA;// This will effect the pattern generated.


  timer = millis(); //sets timer equal to the amount of milliseconds since
the program has began
  pinMode(10, OUTPUT);

}

void loop() {
  if ((millis() - timer) > 500) { //checks if it has been 500 milliseconds
since the last interval
      DisplayColumn++; //increments the collumn
      if (DisplayColumn > 8) { //if the column is above 8 then it will reset
it back to 1
        DisplayColumn = 1;
      }
      if (DisplayRow & 0x0080) { //if bit 7 is high
        DisplayRow = DisplayRow<<1 ; //shift left
      }
      else { //else
        DisplayRow = DisplayRow<<1 ; //shift left
```

```
        DisplayRow |= 0x0001; //force bit 1 to be high
      }
      DisplayRow &= 0x00ff;
      SPI_16 ( ( DisplayColumn <<8 ) + DisplayRow ); //call the spi 16
function

    timer += 500;
  }

}
```