SYNTHIAM

synthiam.com

NeoPixel Ring with Arduino as control board

A NeoPixel ring or strip can be controlled by the EZ-B by using an Arduino as an interface. In this tutorial, we will show the wiring necessary, the Sketch that needs to be loaded onto the Arduino and some example scripts to use in EZ-Builder to use the NeoPixel ring.
Just a big thanks to @RobertL184 (who supplied the original sketch) for the time and energy that he put into getting this working for the community.

Last Updated: 6/17/2017

Wiring

From EZ-B
Ground on D5 to com port ground on Arduino
Signal on D5 to RX on Arduino
Power regulator- This is for the Mini Pro. Yours may be different depending on the arduino. The Neopixel needs 5V though.
5v to Neopixel ring +5v_PWR pin
5v to Arduino 5v pin
ground to Neopixel ring GND pin
ground to Arduino GND pin on Power side of the board
Arduino
Pin D6 to Input Pin on Neopixel Ring
This is the way that I wire mine up. The power for your Arduino may be different. I personally like providing separate power to the NeoPixel instead of coming off of the Arduino. As long as both devices (NeoPixel and the Arduino) are using the same ground, then you will be fine with this wiring description.

Setting up the sketch on the Arduino

At the time of the creation of this tutorial, the current Arduino application version is 1.8.3 and the current version of the NeoPixel library is 1.1.1. This tutorial doesn't cover changes made by Arduino or Adafruit after the writing of this tutorial.
Download and install Arduino IDE software at https://www.arduino.cc/download_handler.php?f=https://www.microsoft.com/store/apps/9nblggh4rsd8?ocid=badge
After the install is complete, launch Arduino IDE and then choose
Sketch from the top menu, then Include Library, then Manage Libraries. The Library Manager will appear on your screen. In the search box at the top right, type in NeoPixel. Chose the item labeled as Adafruit NeoPixel and an Install button will appear. Click the Install button. Once finished, click the close button on the Library Manager.
Copy and Paste this code into a new Sketch and then save it naming it something usefull like NeoPixelRingSerial.

```

/* real time Serial Controlled Neopixel Ring  -- Code not tested yet
/
/ Serial Commands
/     Cr1,g1,b1  - Set whole Ring to same color specified by r1,g1,b1
/     Rt,d  - Rainbow  with interval t and direction d
/     Tr1,g1,b1,r2,g2,b2,t,d - Theater Chase between color r1,g1,b1 and color r2,g2,b2 with interval t and direction d
/     Wr1,g1,b1,t,d - Color Wipe in  color r1,g1,b1 with interval t and direction d
/     Fr1,g1,b1,r2,g2,b2,s,t,d - Fade between color r1,g1,b1 and color r2,g2,b2 in s steps with interval t and direction d
/     Sr1,g1,b1,t  - Scanner with color specified by r1,g1,b1 and interval t
/     Pp,r1,g1,b1 - Set individual pixel p to color specified by r1,g1,b1
/
/    p = pixel number
/    r1 = 1st color red values 0 to 255
/    g1 = 1st color greern values 0 to 255
/    b1 = 1st color blue values 0 to 255
/    r2 = 2nd color red values 0 to 255
/    g2 = 2nd color greern values 0 to 255
/    b2 = 2nd color blue values 0 to 255
/    s = number of steps
/    t = interval time values 0 to 255
/    d = direction  0 = Forward 1 = Backward

*/
#include &ltAdafruit_NeoPixel.h&gt
// Pattern types supported:
enum  pattern { NONE, RAINBOW_CYCLE, THEATER_CHASE, COLOR_WIPE, SCANNER, FADE };
// Patern directions supported:
enum  direction { FORWARD, REVERSE };
// NeoPattern Class - derived from the Adafruit_NeoPixel class
class NeoPatterns : public Adafruit_NeoPixel
{
public:
```

```
// Member Variables:
pattern  ActivePattern;  // which pattern is running
direction Direction;     // direction to run the pattern

unsigned long Interval;   // milliseconds between updates
unsigned long lastUpdate; // last update of position

uint32_t Color1, Color2;  // What colors are in use
uint16_t TotalSteps;  // total number of steps in the pattern
uint16_t Index;  // current step within the pattern

void (*OnComplete)();  // Callback on completion of pattern

// Constructor - calls base-class constructor to initialize strip
NeoPatterns(uint16_t pixels, uint8_t pin, uint8_t type, void (*callback)())
:Adafruit_NeoPixel(pixels, pin, type)
{
    OnComplete = callback;
}

// Update the pattern
void Update()
{
    if((millis() - lastUpdate) > Interval) // time to update
    {
        lastUpdate = millis();
        switch(ActivePattern)
        {
            case RAINBOW_CYCLE:
                RainbowCycleUpdate();
                break;
            case THEATER_CHASE:
                TheaterChaseUpdate();
                break;
            case COLOR_WIPE:
                ColorWipeUpdate();
                break;
            case SCANNER:
                ScannerUpdate();
                break;
            case FADE:
                FadeUpdate();
                break;
            default:
                break;
        }
    }
}

// Increment the Index and reset at the end
void Increment()
{
    if (Direction == FORWARD)
    {
       Index++;
       if (Index >= TotalSteps)
        {
            Index = 0;
            if (OnComplete != NULL)
            {
                OnComplete(); // call the comlpetion callback
            }
        }
    }
    else // Direction == REVERSE
    {
        --Index;
        if (Index > 1, Green(color) >> 1, Blue(color) >> 1);
    return dimColor;
}
```

```
// Set all pixels to a color (synchronously)
void ColorSet(uint32_t color)
{
    for (int i = 0; i > 16) & 0xFF;
}

// Returns the Green component of a 32-bit color
uint8_t Green(uint32_t color)
{
    return (color >> 8) & 0xFF;
}

// Returns the Blue component of a 32-bit color
uint8_t Blue(uint32_t color)
{
    return color & 0xFF;
}

// Input a value 0 to 255 to get a color value.
// The colours are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos)
{
    WheelPos = 255 - WheelPos;
    if(WheelPos  0) {
inchar = Serial.read();
if ( inchar == 'C') {    // string should start with C
  red1 = Serial.parseInt();     // then an ASCII number for red
  green1 = Serial.parseInt();   // then an ASCII number for green
  blue1 = Serial.parseInt();    // then an ASCII number for blue
  Ring1.ActivePattern = NONE;
  Ring1.ColorSet(Ring1.Color(red1, green1, blue1));
}
if ( inchar == 'R') {    // string should start with R
  t = Serial.parseInt();     // then an ASCII number for interval
  d = Serial.parseInt();    // then an ASCII number for direction
  if (d ==0) {
     Ring1.RainbowCycle(t, FORWARD);
  } else {
     Ring1.RainbowCycle(t, REVERSE);
  }
}
if ( inchar == 'F') {    // string should start with F
  red1 = Serial.parseInt();     // then an ASCII number for red
  green1 = Serial.parseInt();   // then an ASCII number for green
  blue1 = Serial.parseInt();    // then an ASCII number for blue
  red2 = Serial.parseInt();     // then an ASCII number for red
  green2 = Serial.parseInt();   // then an ASCII number for green
  blue2 = Serial.parseInt();    // then an ASCII number for blue
  s = Serial.parseInt();        // then an ASCII number for steps
  t = Serial.parseInt();        // then an ASCII number for interval
  d = Serial.parseInt();        // then an ASCII number for direction
  if (d ==0) {
    Ring1.Fade(Ring1.Color(red1, green1, blue1), Ring1.Color(red2, green2, blue2), s, t,
FORWARD);
  } else {
    Ring1.Fade(Ring1.Color(red1, green1, blue1), Ring1.Color(red2, green2, blue2), s, t,
REVERSE);
  }
}
if ( inchar == 'T') {    // string should start with T
  red1 = Serial.parseInt();     // then an ASCII number for red
  green1 = Serial.parseInt();   // then an ASCII number for green
  blue1 = Serial.parseInt();    // then an ASCII number for blue
  red2 = Serial.parseInt();     // then an ASCII number for red
  green2 = Serial.parseInt();   // then an ASCII number for green
  blue2 = Serial.parseInt();    // then an ASCII number for blue
  t = Serial.parseInt();        // then an ASCII number for interval
  d = Serial.parseInt();        // then an ASCII number for direction
  if (d ==0) {
```

```
    Ring1.TheaterChase(Ring1.Color(red1, green1, blue1), Ring1.Color(red2, green2, blue2), t,
FORWARD);
  } else {
    Ring1.TheaterChase(Ring1.Color(red1, green1, blue1), Ring1.Color(red2, green2, blue2), t,
REVERSE);
  }
}
if ( inchar == 'W') {    // string should start with W
  red1 = Serial.parseInt();     // then an ASCII number for red
  green1 = Serial.parseInt();   // then an ASCII number for green
  blue1 = Serial.parseInt();    // then an ASCII number for blue
  t = Serial.parseInt();        // then an ASCII number for interval
  d = Serial.parseInt();        // then an ASCII number for direction
  if (d ==0) {
    Ring1.ColorWipe(Ring1.Color(red1, green1, blue1), t, FORWARD);
  } else {
    Ring1.ColorWipe(Ring1.Color(red1, green1, blue1), t, REVERSE);
  }
}
if ( inchar == 'S') {    // string should start with S
  red1 = Serial.parseInt();     // then an ASCII number for red
  green1 = Serial.parseInt();   // then an ASCII number for green
  blue1 = Serial.parseInt();    // then an ASCII number for blue
  t = Serial.parseInt();        // then an ASCII number for interval
  Ring1.ColorWipe(Ring1.Color(red1, green1, blue1), 50, FORWARD);
  Ring1.ColorSet(Ring1.Color(0,0, 0));
  Ring1.Scanner(Ring1.Color(red1, green1, blue1), t);
 }
 if ( inchar == 'P') {    // string should start with P
  my_pixel = Serial.parseInt();     // then an ASCII number for pixwel
  red1 = Serial.parseInt();     // then an ASCII number for red
  green1 = Serial.parseInt();   // then an ASCII number for green
  blue1 = Serial.parseInt();    // then an ASCII number for blue
  Ring1.ActivePattern = NONE;
  Ring1.setPixelColor(my_pixel, Ring1.Color(red1, green1, blue1));
  Ring1.show();
}
```

```
}
// Update the ring.
Ring1.Update();
}
// Ring1 Completion Callback
void Ring1Complete()
{
if ((Ring1.ActivePattern == RAINBOW_CYCLE) || (Ring1.ActivePattern == FADE) ||
(Ring1.ActivePattern == THEATER_CHASE) || (Ring1.ActivePattern == SCANNER)){
Ring1.Reverse();
}
}
```

Send this code to your Arduino after connecting your Arduino to the computer. How to do this is dependent on your Arduino and is beyond the scope of this tutorial.

Commands sent from the EZ-B to the Arduino

The top portion of the sketch contains all of the commands that the Arduino can accept. Some commands take more parameters than others. Below are the comments that are in the script which outline each command and its needed parameters.

```

/ Serial Commands
/     Cr1,g1,b1  - Set whole Ring to same color specified by r1,g1,b1
/     Rt,d  - Rainbow  with interval t and direction d
/     Tr1,g1,b1,r2,g2,b2,t,d - Theater Chase between color r1,g1,b1 and color r2,g2,b2 with interval t and direction d
/     Wr1,g1,b1,t,d - Color Wipe in  color r1,g1,b1 with interval t and direction d
/     Fr1,g1,b1,r2,g2,b2,s,t,d - Fade between color r1,g1,b1 and color r2,g2,b2 in s steps with interval t and direction d
/     Sr1,g1,b1,t  - Scanner with color specified by r1,g1,b1 and interval t
/     Pp,r1,g1,b1 - Set individual pixel p to color specified by r1,g1,b1
/
/    p = pixel number
/    r1 = 1st color red values 0 to 255
/    g1 = 1st color greern values 0 to 255
/    b1 = 1st color blue values 0 to 255
/    r2 = 2nd color red values 0 to 255
/    g2 = 2nd color greern values 0 to 255
/    b2 = 2nd color blue values 0 to 255
/    s = number of steps
/    t = interval time values 0 to 255
/    d = direction  0 = Forward 1 = Backward

*/

```

If you used the wiring section that I provided, you should be able to use this example script in EZ-Builder to run through a few of the different modes for the NeoPixel.
This command will send a serial command from digital pin 5 at 9600 baud of "C100,1,1" which will turn the ring red.

```

sendserial( D5, 9600, "C100,1,1")

```

This command will send a serial command from digital pin 5 at 9600 baud of "R20.0" which will make a rainbow effect.

```

sendserial( D5, 9600, "R20,0")

```

This command will send a serial command from digital pin 5 at 9600 baud of "T100,1,1,1,100,1,20,0" which will make a a theater chase effect.

```

sendserial( D5, 9600, "T100,1,1,1,100,1,20,0")

```

With these examples, you should be able to see how the parameters are passed and used. To turn off the NeoPixel, just send a command of C0,0,0 to the controller. This sets all pixels to off.

```

sendserial( D5, 9600, "C0,0,0")

```

If you have more or less pixels that 24, or you use a different pin on the Arduino to connect to the NeoPixel

If you want to change the pin that the arduino is connecting to the NeoPixel on, change the 2nd parameter in line 321 in the sketch. If you want to change the number of NeoPixles in the ring or device, change the first parameter in line 321.
Example

```

NeoPatterns Ring1(24, 6, NEO_GRB + NEO_KHZ800, &Ring1Complete);

```

is for a 24 pixel NeoPixel that has its input pin connected to pin 6 on the Arduino.