Light of Your Life Wedding Bouquet
Created by Erin St Blaine

Last updated on 2021-10-22 11:15:18 AM EDT
## Guide Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guide Contents</td>
<td>2</td>
</tr>
<tr>
<td>Introduction</td>
<td>3</td>
</tr>
<tr>
<td>Materials</td>
<td>3</td>
</tr>
<tr>
<td>For Additional Lighting (optional)</td>
<td>4</td>
</tr>
<tr>
<td>For the Bouquet</td>
<td>4</td>
</tr>
<tr>
<td>Planning and Testing</td>
<td>5</td>
</tr>
<tr>
<td>Feather Setup</td>
<td>5</td>
</tr>
<tr>
<td>Pixel Testing Setup</td>
<td>6</td>
</tr>
<tr>
<td>Flower Setup</td>
<td>7</td>
</tr>
<tr>
<td>Wiring Diagram</td>
<td>9</td>
</tr>
<tr>
<td>Feather &amp; Neopixel Assembly</td>
<td>10</td>
</tr>
<tr>
<td>Software</td>
<td>12</td>
</tr>
<tr>
<td>Testing &amp; Tweaking your Code</td>
<td>16</td>
</tr>
<tr>
<td>Variables to Play With</td>
<td>16</td>
</tr>
<tr>
<td>Additional Lighting Assembly</td>
<td>17</td>
</tr>
<tr>
<td>On/Off Switch</td>
<td>17</td>
</tr>
<tr>
<td>More Lights</td>
<td>17</td>
</tr>
<tr>
<td>String your Neopixels</td>
<td>18</td>
</tr>
<tr>
<td>Fiber Optics &amp; Warm White LEDs</td>
<td>18</td>
</tr>
<tr>
<td>Glowbys</td>
<td>18</td>
</tr>
<tr>
<td>Warm White LEDs</td>
<td>19</td>
</tr>
<tr>
<td>Putting it All Together</td>
<td>20</td>
</tr>
<tr>
<td>Build the Boutonniere</td>
<td>22</td>
</tr>
<tr>
<td>Build the Bouquet</td>
<td>25</td>
</tr>
</tbody>
</table>
She lights up the room when she walks in. Make sure everyone can see it.

Two Feather 32u4 Radio modules are hidden inside a bouquet and boutonniere along with a few discreet neopixels.

Using the radio signal strength measurements, the two Feathers can roughly determine how far apart they are. When the two get close together, the lights glow brighter, and when they separate the lights dim and flicker!

As she walks down the aisle, her bouquet will slowly increase in brightness until she's standing right by his side, and beaming.

This technique will work great for any two costume pieces as well. Find your partner at the after-dark party by following the brightness of your light-up hat or other accessory.

Materials

- [Two Feather 32u4 RFM69HCW (I used 915MHz)](https://adafruit.it/pC8)
- [Neopixels](http://adafruit.it/1559)
- [30awg stranded wire](http://adafruit.it/2051)
- [2 On/off switches](http://adafruit.it/1092)
• 2 Battery extension cables (http://adafruit.it/1131)
• 2 LiPoly batteries - one 350mAh (http://adafruit.it/2750) and one 1200mAh (http://adafruit.it/258)

For Additional Lighting (optional)

• Warm white LED strand (http://adafruit.it/893)
• Glowby fiber optic barette (https://adafruit.it/ofT)
• 2-3 resistors: 20-100ohm

For the Bouquet

• Flowers (real or silk will work)
• Floral tape
• Ribbons & decorations
• Floral Pins
Planning and Testing

Feather Setup

The hardware build for this project is fairly simple and straightforward. The software part is a little trickier and requires a bit of setup.

If you're a newbie to Arduino, start with this guide (https://adafruit.it/pC9).

If you have never used a Feather, start with this guide (https://adafruit.it/ral).

If you're not comfortable installing libraries, here's the guide for you (https://adafruit.it/dit).

Be sure you have the Feather boards set up and working (be sure you can run the "Blink" sketch) before proceeding.
Pixel Testing Setup

Whenever you work with neopixels, testing at each step means you'll catch any shorts or mistakes early. Plus you get to see your lights come on right away which makes the whole process much more fun and satisfying.

My favorite way to test neopixels is using a Gemma microcontroller (http://adafru.it/1222) and some alligator clips. The Gemma is inexpensive and really easy to use for prototyping. You can test any combination of wires without soldering or mucking about with breadboards and headers.

You'll need to make sure you have Adafruit's board support installed as well as the Adafruit Neopixel library (https://adafru.it/nBF).

Plug your Gemma in to your computer using its USB port. Open your Arduino IDE and select Adafruit Gemma from your Boards menu.

Go to File > Examples > Adafruit_Neopixel > strandtest and open the strandtest code. Find this line at the top:

```cpp
#define PIN 6
```

Change PIN to 1 instead of 6.

Press the Reset button on the Gemma to get it into bootloader mode, and then immediately press the upload button in Arduino to upload the code.
Then, get your alligator clips out and hook them up thusly:

<table>
<thead>
<tr>
<th>Wire Color</th>
<th>Gemma Pin</th>
<th>NeoPixel Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>G</td>
<td>–</td>
</tr>
<tr>
<td>Red</td>
<td>Vout</td>
<td>+</td>
</tr>
<tr>
<td>Yellow</td>
<td>D1</td>
<td>Arrow pointing toward LED</td>
</tr>
</tbody>
</table>

The clips' colors will correspond to the wire colors used in the rest of this project. You can power the Gemma from the USB port or plug a battery in to the JST connector. As you're soldering pixels together, hook each one up to the Gemma to be sure it comes on and shows all the colors.

Tip: if testing NeoPixels on a “live” circuit, *always* connect the ground (black) wire first, and disconnect it last. There's a small chance of damaging a NeoPixel otherwise.

Flower Setup

If you're working with silk or latex flowers, your job is easy. Choose the ones you like. Remember that silk flowers go "out of season" at local craft stores just like real flowers do, so give yourself time to order online if you want specific flowers.

A good search term is "Real Nature Touch" -- these are good quality latex flowers that feel and look almost like the real thing.
I used real flowers, but still picked up a few latex ones just to experiment on.

Calla lilies are wonderful at hiding neopixels. I ordered my flowers from CalCallas.com (https://adafruit.com/pCb). Plan to have your flowers delivered a couple days before your event -- fresh, quality flowers will open up just a little more in those last few days and your bouquet will be at its peak. And remember, you don’t want to be rushing to finish it on the day of the wedding.

I ordered over the phone and they put together a custom 20-flower order for me with Picasso purple, white and yellow flowers, for a lot less money than ordering a pre-assembled bouquet. My finished bouquet had 17 flowers, leaving one for the boutonniere and two for a Mother-of-the-Groom boutonniere.
Wiring Diagram

You will be assembling two nearly-identical circuits.

The jumper wire (shown in orange) between pins 9 and 10 should be installed on only ONE of the two boards! It won't work any other way. This distinguishes the "transmitter" board from the "receiver" board.

The green wire is an antenna...required on both boards...it needs to be a specific length, explained further on the next page...
Feather & Neopixel Assembly

Cut two lengths of green wire to 3 inches exactly, and solder it into the ANT pin on each of your Feather boards.

More about antenna options (https://adafruit.it/pCc)

Get out two neopixels and solder a red wire to +, a black wire to -, and a white wire to "in". Test the pixels with your Gemma tester to make sure they come on and light up in all the colors.

Solder each neopixel to a feather board:
- Red wire to BAT
- Black wire to G
- White wire to 6
On ONE of the feather boards, add a small jumper wire in pins 9-10. This jumper will be referenced in the code to tell the two boards apart (transmitter vs receiver).

Once your soldering is done and all your connections are clean, it's time to upload the code to the board and refine your variables.
Software

Start by installing these libraries in your Arduino IDE:

https://adafruit.it/tXd
https://adafruit.it/tXd

https://adafruit.it/pCe
https://adafruit.it/pCe

You can also install FastLED directly within the Arduino IDE (Sketch > Include Library > Manage Libraries...)

Got everything installed? Grab the code below and paste it into the Arduino IDE. If you have issues check out the Feather 32u4 HELP page (https://adafruit.it/rcU)

```cpp
// Distance meter using the RSSI function of two RFM69 radios.
// This is a very VERY "ish" gauge of *relative* distance (not actual units
// like feet or meters or anything), and is prone to interference from
// walls, bodies, direction and the like...it's intended for an artistic
// presentation, nothing scientific.

#include <RFM69.h> // get it here: https://www.github.com/lowpowerlab/rfm69
#include <SPI.h>
#include "FastLED.h"

// RADIO HARDWARE CONFIG -----------------------------------------------
#define TX_NODE_ID   1
#define RX_NODE_ID   2
#define NETWORK_ID 100 // The same on all nodes that talk to each other
#define ENCRYPTKEY "sampleEncryptKey" // Same 16 characters on all nodes!

// Match frequency to the hardware version of the radio on your Feather:
//#define FREQUENCY   RF69_433MHZ
//#define FREQUENCY   RF69_868MHZ
#define FREQUENCY   RF69_915MHZ
#define IS_RFM69HCW true // Set true if using RFM69HCW module

// RFM69 pins on Feather 32U4 board:
#define RFM69_CS   8
#define RFM69_IRQ  7
#define RFM69_IRQN 4
#define RFM69_RST  4

// A jumper across two pins selects whether this board is the
// TRANSMITTER or the RECEIVER. That way the SAME code can be uploaded
// to BOTH boards. I got tired of having to comment out or enable a
// line every time I uploaded to one or the other.
```

© Adafruit Industries  https://learn.adafruit.com/light-of-your-life-wedding-bouquet
#define TX_SENSE_1  9
#define TX_SENSE_2 10

RFM69 radio(RFM69_CS, RFM69_IRQ, IS_RFM69HCW, RFM69_IRQN);
bool isTransmitter;

// SIGNAL FILTERING CONFIG ---------------------------------------------
#define MEDIAN_SIZE  5 // Number of RSSI readings to be median filtered
#define AVERAGE_SIZE 8 // Number of median results to be averaged

// Signal values are stored as abs(RSSI), hence the positive values here:
#define SIGNAL_MIN   26 // abs() of signal strength at closest distance
#define SIGNAL_MAX  100 // furthest distance
#define SIGNAL_INIT  ((SIGNAL_MAX + SIGNAL_MIN) / 2 - SIGNAL_MIN)

uint8_t medianBuf[MEDIAN_SIZE], // Prior abs(RSSI) readings
    averageBuf[AVERAGE_SIZE], // Prior medians
    medianIdx = 0, // Current position in medianBuf[]
    averageIdx = 0; // Current position in averageBuf[]

uint32_t sum = SIGNAL_INIT * AVERAGE_SIZE + (AVERAGE_SIZE / 2);

// DISPLAY CONFIG -----------------------------------------------
#define LED_TYPE    WS2812
#define DATA_PIN  6 // NeoPixels are connected to this pin
#define NUM_LEDS 20 // How many leds in your strip?
CRGB leds[NUM_LEDS]; //create your NeoPixel array
#define COLOR_ORDER GRB

int led = 13;
uint8_t average = 10;
uint8_t distance = 15; // Play with this number to affect the range

CRGBPalette16 currentPalette;
TBlendType currentBlending;

int HUE = 60;
int SATURATION = 255;
int BRIGHTNESS = 255;

#define FPS 25 // Animation rate (frames per second)

// SETUP FUNCTION - runs once at startup ---------------------------------

void setup() {
    // pinMode(led, OUTPUT); // Turns the onboard LED on -- useful for testing.
    // digitalWrite(led, HIGH); // Comment out these two lines if you want it off while running

    // Un-comment these 3 lines to refine your signal variables ---------
    // while (!Serial) {}  
    // Serial.begin(9600);
    // Serial.println("hello");

    // Set up jumper detect on TX_SENSE pins
    pinMode(TX_SENSE_2, OUTPUT);
    digitalWrite(TX_SENSE_2, LOW);
    pinMode(TX_SENSE_1, INPUT_PULLUP):
}
// Reset the RFM module:
pinMode(RFM69_RST, OUTPUT);
digitalWrite(RFM69_RST, HIGH);
delay(100);
digitalWrite(RFM69_RST, LOW);
delay(100);

// Test for jumper on TX_SENSE pins
isTransmitter = digitalRead(TX_SENSE_1);

// Initialize radio:
radio.initialize(FREQUENCY, isTransmitter ? TX_NODE_ID : RX_NODE_ID);
if(IS_RFM69HCW) radio.setHighPower(); // Only for RFM69HCW & HW!
radio.setPowerLevel(31); // Output range 0 (5dBm) to 31 (20dBm)
radio.encrypt(ENCRYPTKEY);

memset(medianBuf, SIGNAL_INIT, sizeof(medianBuf));
memset(averageBuf, SIGNAL_INIT, sizeof(averageBuf));

// Set up a timer interrupt for LED animation. This ensures uniform
// timing for animation while radio ACK time may be unpredictable.
// THIS CODE IS SPECIFIC TO ATMEGA32U4; will not run on M0, Teensy3, etc.
TCCR1A = _BV(WGM11) | _BV(WGM10);                         // Fast PWM mode
TCCR1B = _BV(WGM13) | _BV(WGM12) | _BV(CS11) | _BV(CS10); // 1:64 prescale
OCR1A = ((F_CPU / 64) / FPS) - 1;
TIMSK1 = _BV(TOIE1); // Enable overflow interrupt on Timer/Counter 1

// tell FastLED about the LED strip configuration
FastLED.addLeds<LED_TYPE, DATA_PIN,COLOR_ORDER>(leds, NUM_LEDS)
  .setCorrection(TypicalLEDStrip)
  .setDither(BRIGHTNESS < 255);
  currentBlending = LINEARBLEND;
}

// LOOP FUNCTION - runs repeatedly -----------------------------------------

void loop() {

  if(isTransmitter) {
    // Send a small message to the receiver node...
    radio.sendWithRetry(RX_NODE_ID, "<3", 2);
  } else {
    // Receiver waits for message, sends acknowledgement.
    // The ACK is needed to gauge distance on BOTH ends.
    radio.receiveDone();
    if(radio.ACKRequested()) radio.sendACK();
  }

  if(radio.RSSI) { // Got signal strength?
    uint16_t s = abs(radio.RSSI);
    if(s < SIGNAL_MIN) s = SIGNAL_MIN;
    else if(s > SIGNAL_MAX) s = SIGNAL_MAX;
    s -= SIGNAL_MIN; // s is now 0 to (MAX-MIN)

    medianBuf[medianIdx] = s; // Store new value in median buffer
    if(++medianIdx >= MEDIAN_SIZE) medianIdx = 0;
  }
}
// Median filter discards 'outliers,' the most spurious readings;
// abrupt changes that are possibly inaccurate.

// Make a sorted copy of median buffer to find actual current median
// without disturbing original values. Since it's just a few elements,
// a crude insertion sort is done...
uint8_t m[MEDIAN_SIZE], median, average, i, j, k;
memcpy(m, medianBuf, sizeof(m));
for(i=1; i<MEDIAN_SIZE; i++) {
  k = m[i];
  for(j=i; j && (m[j-1] > k); j--) m[j] = m[j-1];
  m[j] = k;
}
median = m[MEDIAN_SIZE / 2];

// Following median, the average of prior results is taken,
// further smoothing out any jumpy bits...
sum -= averageBuf[averageIdx];
averageBuf[averageIdx] = median;
sum += median;
if(++averageIdx >= AVERAGE_SIZE) averageIdx = 0;
average = sum / AVERAGE_SIZE;

// 'average' is the signal strength from the radio...
// it'll likely be between 0 (closest) and 75 or so
// (furthest...like, down the block).

// UNCOMMENT THESE SERIAL COMMANDS to see what your radios are sending while the Feather is
// plugged in via USB
//Serial.print("Average ");
//Serial.println(average);
//Serial.print("Adjusted ");
//Serial.println(300-(average*distance));
//Serial.print("Brightness ");
//Serial.println(constrain(map ((300-(average*distance)),-400,300,0,255),10,255));

//This line is your main calculation. It's taking "average" (your radio signal, averaged
//out for smoothness) and
// multiplying it by "distance" (currently set at 15, but play with that number to change the
//range),
// and then constraining it between a brightness of 10 (dimmest setting) and 255 (brightest
//setting).

BRIGHTNESS=(constrain(map ((300-(average*distance)),-400,300,0,255),10,255));
}
ISR(TIMER1_OVF_vect) { // Timer 1 interrupt for animation
  // Display *prior* frame of data at start of interrupt -- this ensures
  // uniform timing for effects that may have variable processing times.
  FastLED.show();

  // Render one frame of animation here...pick one function, comment out other
  Solid();
  //Colors();
Testing & Tweaking your Code

Your environment or particular antenna may be different from mine. Try uploading the code as-is and watch the behavior of the LEDs as you move them closer and further apart. Blocking the signal with your body will also change your results, as will walls, trees, or excited grandmothers.

Variables to Play With

Change the "distance" variable up or down to play with the range over which the brightness changes.

If they don't seem as responsive as you'd like, try changing the MEDIAN_SIZE and AVERAGE_SIZE values.

Look for "serial" commands in the code (currently all commented out). If your LEDs aren't performing the way you want, uncomment these lines to watch what's happening with your numbers in your serial monitor. Just be sure to comment them out again or your boards won't work without being plugged in.

Give the HUE and SATURATION variables a number between 0 and 255 to choose your LED color. Or choose multiple colors by using "Colors();" instead of "Solid();" and assigning different colors to different pixels.

Make sure the two boards are working and communicating before continuing. If no lights come on... aside from the usual checking for electrical shorts, etc., make sure you have the jumper wire installed between pins 9 and 10 on just one of the two boards.
Additional Lighting Assembly

On/Off Switch

At the time this guide was being written, it was necessary to cobble together two pieces to create a battery switch. We now have a JST extension cable with built-in switch! (http://adafru.it/3064) If using the newer part, you can skip ahead to the “More Lights” section below. Otherwise, if you already have these components in your stash, or want something ever-so-slightly smaller, here’s how the two-part solution was made:

A small battery switch is made from our Tactile On/Off Switch with Leads (http://adafru.it/1092) and a JST-PH Battery Extension Cable (http://adafru.it/1131).

Trim the leads on your tactile on/off switches to around 2 inches.

Cut both ends from your battery extension cables, with about 1 inch of wire attached. Solder the ground (black) wires together. Solder the switch in between the two power (red) wires. This will plug into your battery and directly into the Feather. Insulate the solder connections with heat-shrink tubing to prevent electrical shorts.

You’ll need two of these; one each for the bouquet and boutonniere.

More Lights

Now it's time to add more neopixels, fiber optics, and light strands to make your bouquet into something truly magical.

A good design has texture and variation. I wanted three separate types of lighting in my final bouquet, so I added a modified glowby hair barette for fiber optics and a warm white LED strand.

For my bouquet, I used 17 neopixels, a fiber optic glowby and a warm white LED strand.
For my boutonniere I used just one neopixel and a fiber optic glowby.

String your Neopixels

Here’s a silly video showing my favorite technique for making a neopixel strand. Remember to leave enough wire between each pixel to thread into the flower and back out again -- for calla lilies, 6-8 inches of wire seemed to be about right.

Fiber Optics & Warm White LEDs

One option is to keep the build simple and just add these elements in as they are, and use the coin cell batteries to power them. But with a little more soldering and a couple resistors, you can hook them up to the same battery and on/off switch as the Feather and neopixels. That way the bride and groom just have one button to think about, instead of having to fiddle with three separate switches.

Glowbys
Open the glowby and straighten out the two LED leads. Cut off any extraneous plastic.

Be sure you know which is the positive and which is the negative lead (test it with a coin cell battery).

Solder a resistor onto your positive lead. I used a 33 ohm resistor, which gave me maximum brightness without blowing out the LED. Test and be sure it's working with your Gemma tester.

Warm White LEDs
Cut the battery pack off the warm white LED strand (cut one wire at a time to avoid damaging your lights). Scrape the wires with a utility knife to remove the clear coating.

Use a coin cell battery to determine which is the + and which is the - lead.

Solder a resistor onto one of the leads. Experiment to find the right brightness. I found that a 33 ohm resistor was just right to balance the neopixels.
Carefully un-solder the red and black neopixel wires from your Feather boards.

Twist all three power wires and all three ground wires together.

Solder the combined power wires into BAT on the feather, and the combined ground wires into G.
Build the Boutonniere

Time to play with flowers! Get out your floral tape and wrap your neopixel, wires, and fiber optics like a mummy.

Floral tape is wonderful stuff. It sticks to itself without being sticky, and lets the light shine through beautifully while covering up the black electronics.

Floral Tape FTW!! Who Knew??
I may just use floral tape in every electronics project ever from now on.

Slide your neopixel inside your main flower. Choose a couple more flowers and tape them all together.

Tape the battery and the feather to the stems, with the USB port at the bottom.

Wrap all the wires neatly, with the USB port available for charging, the switch available for switching, and the green antenna wire coming out the top.

If you're making this a few days ahead of time, leave the flower stems extra long so you can keep it in a vase of water without submerging the electronics.

Finish by wrapping the base with ribbon to cover the floral tape.
Build the Bouquet

Wrap the neopixels and any other wiring in floral tape.

Arrange the flowers the way you want them, and slip the neopixels inside each one. This is a good time to go back to your code and reorganize the colors of each pixel so it complements its flower.
Wrap floral tape around your battery and feather board, with the switch and antenna coming out the top and the USB port accessible at the bottom.

Add in any additional flowers, leaves, or accents and arrange the fiber optics between the flowers.

Tape the electronics to the flower stems. Finish by weaving the LED strand through the stems and flowers, and wrap the whole thing in pretty ribbon to cover the floral tape.

To charge the batteries, **be sure the bouquet is switched ON**, then plug in the USB cable. (It won't charge if it's turned off)

Let them charge **completely** before the wedding. Overnight is good. Also bring a spare battery for each, and some floral tape and ribbon and pins for emergency repairs. Murphy's Law holds extra power on your wedding day, and you don't want to find that the batteries didn't charge and the whole shebang doesn't
work.