ISS Pin

Created by Leslie Birch

https://learn.adafruit.com/iss-pin

Last updated on 2023-08-29 03:14:03 PM EDT

Table of Contents

Overview 3

Tools & Supplies 3

Solder Circuit 4

Create Cover 5

Code 6

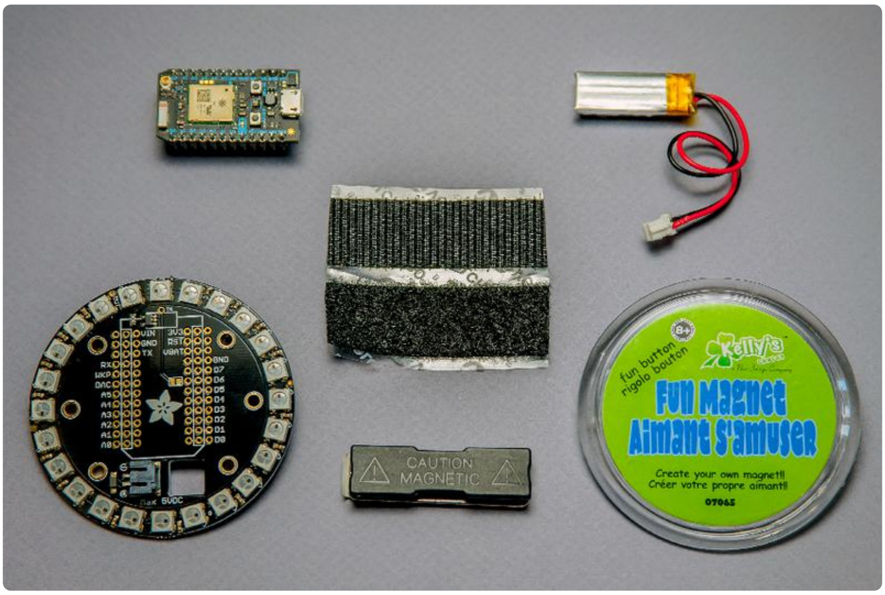
Set Up IFTTT 9

Wear It! 11

Overview

A Particle Photon microcontroller and an Adafruit Neopixel ring combine to make a pin that's fit for NASA fans. It displays an orbiting white blip when idle and then turns blue, white, red and multicolor when the ISS flies by. The code takes advantage of the Photon's Wi-Fi capability and uses IFTTT (If This Then That), a free site that makes connecting IoT devices as easy as a few clicks. This project was inspired by my first NASA Space Apps Challenge project created with friend Brooks Zurn Rampersad--the ISS Orbit Skirt. That's enough history...on to the parts!

Tools & Supplies



Particle Photon and Adafruit's Particle/Spark NeoPixel Ring make a great combo for this pin, creating the perfect platform for a planet. Notice the plastic button which allows for any photo insert. Although I've included a battery, that's really just for showing off to a class or friends for an hour. You'll really need to have a USB battery pack and cable if you want to wear it in your fave wifi zone for an extended period of time.

Supplies:

- Particle Photon With Headers ()
- Adafruit Particle/Spark NeoPixel Ring Kit ()
- USB Cable A/MicroB ()
- LiPoly Battery ()(optional)
- Adafruit LiPo Micro USB Charger () (optional)
- USB Battery Pack () (you probably have one at home)

- Magnetic Pin Back ()
- Adhesive Velcro (found in craft stores)
- Black Construction Paper
- White Paper (decent stock for printing Earth image)
- Solder

Tools:

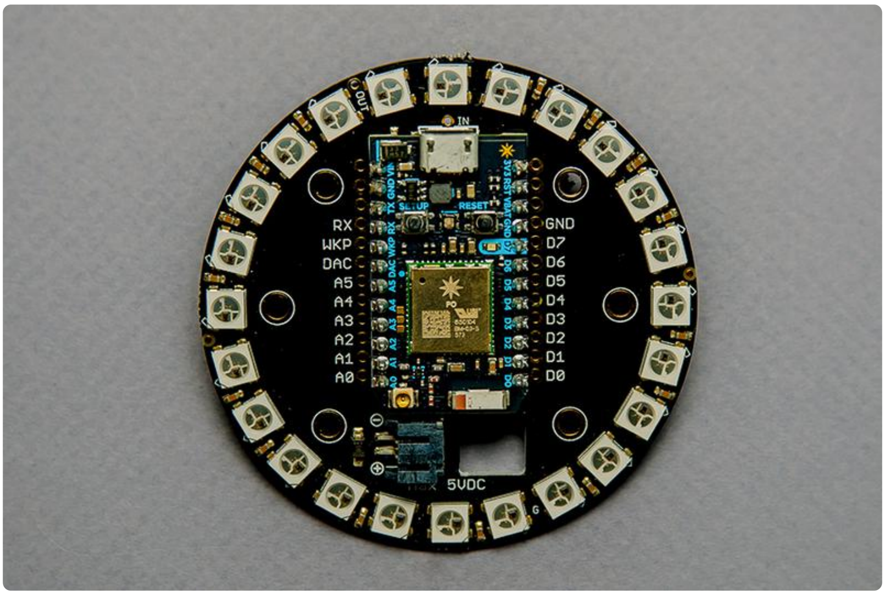
- Soldering Iron (with small tip)
- Diagonal Snips

Make sure you've taken the time to thoroughly go over the guides for:

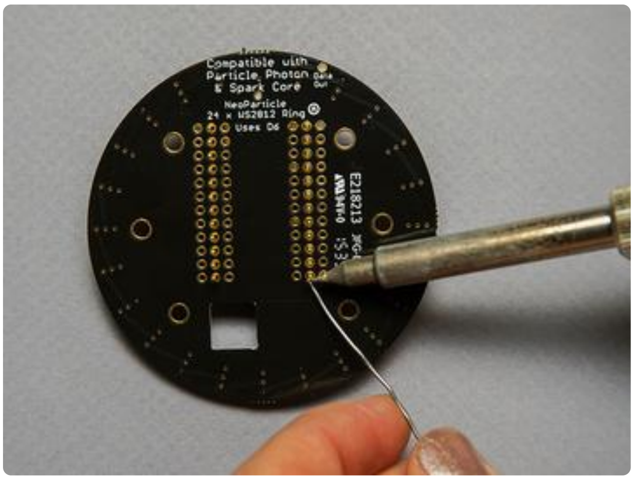
- Particle Photon () (on Particle's site)
- Adafruit NeoPixel UberGuide ()

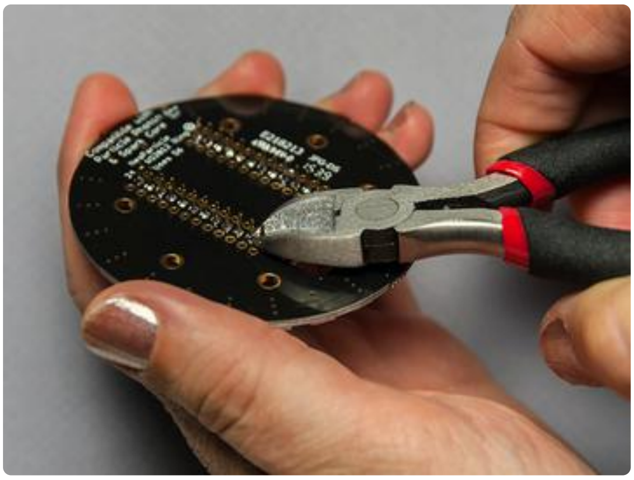
Make sure you have fully registered and tested your Particle Photon before doing this project. Paraticle.io has its own IDE for code.

Solder Circuit



Insert the pins of the Photon through the NeoPixel ring board so there is a row of holes on the outside surrounding the microcontroller (basically aligning the microcontroller with the outline on the board).



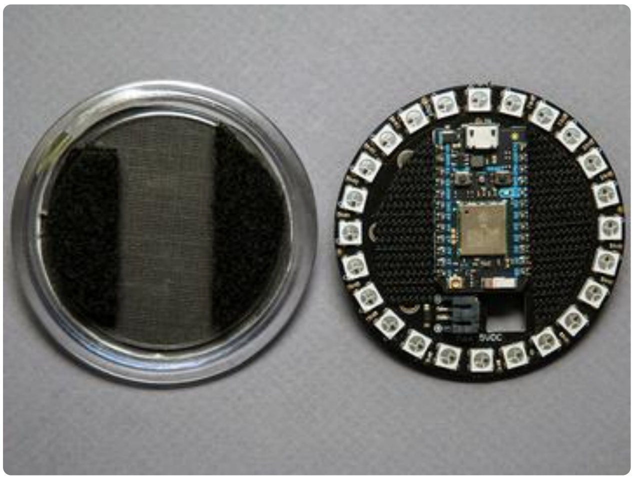


Solder all pins on the reverse side being careful not to create bridges since the holes are close together. After soldering, snip the remaining length of the pins so you have a smooth back for your pin.

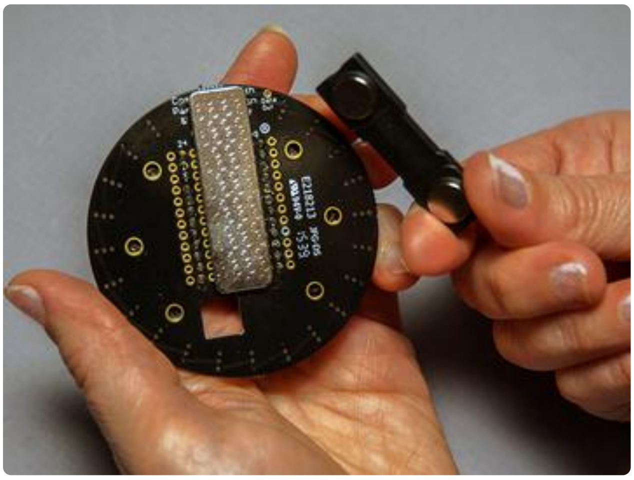
Create Cover

Take the plastic button and remove the magnet from the back. We'll need it to be smooth to lay on top of the microcontroller. Remember we have a magnet pin back we'll be placing on the back of the NeoPixel ring.

Go to NASA's photo library () and select your favorite Earth image. Size it to fit your button (mine was about 2 1/8 in.) and print it on paper/card stock. Cut out the image, as well as a matching circle of black paper. Then, pry open the plastic back of the button and place both the image and black paper inside with Earth facing out.



Now in order to create a removable hatch, we'll need to build up the inside area between the plastic button and microcontroller with velcro. You can create 2-3 layers like I did on either side of the Photon, or you can choose to glue some foam core or cardboard on the bottom layer with some velcro on top. The trick is to get the button to lay level on the top of the Photon as the micro jack makes it slightly uneven. Test out your layers of velcro before adhering them.



The finishing touch is to add the magnetic pin on the back of the NeoPixel ring board. Just center it length-wise between the holes on the board. This is a strong pin so you won't have to worry about it falling off.

Code

Make sure your Photon is plugged into your computer and sign into Particle.io (). Then, go into "Console" and make sure you've selected your Photon device. Next, go into the "IDE" and enter the code below. It's important to load the code before you get to the next step of "Finishing Touches" as a function is called that needs to be read by the program IFTTT (IfThisThenThat).

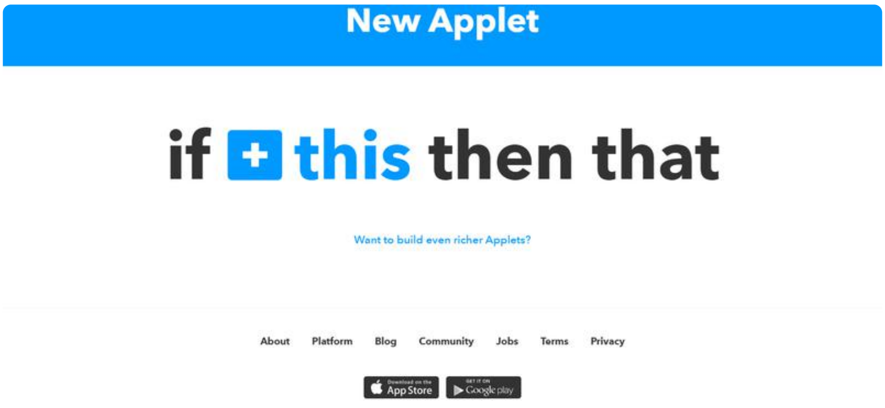
Make sure you have added the Adafruit NeoPixel Library in Particle Build for this sketch to work. Just because the code lists the library doesn't mean that it is seen. It will generate its own line of code.

```
/* ISS Pin by Leslie Birch for Adafruit Industries
Based on NeoPixel Library by Adafruit and code from Particle.io
This version connects with IFTTT and shows a ring of blue, white, red and multi
when triggered.
When not triggered, it does a comet chase of white. */

// This #include statement was automatically added by the Particle IDE.
#include &lt;neopixel.h&gt;

#include "application.h"

SYSTEM_MODE(AUTOMATIC);

void colorWipe(uint32_t c, uint8_t wait);

#define PIXEL_PIN D6
#define PIXEL_TYPE WS2812B

const int FADE_LENGTH = 3;
const int FADE_SCALE  = 128;
const int PIXEL_COUNT = 24;

Adafruit_NeoPixel strip = Adafruit_NeoPixel(PIXEL_COUNT, PIXEL_PIN, PIXEL_TYPE);

bool toggle = false;

int pos = 0, dir = 1;
int j;

int pixelToggle(String command);

void setup(){
  strip.begin();
  strip.setBrightness(70); //set the brightness of NeoPixels to your preference
  strip.show(); // Initialize all pixels to 'off'

    Particle.function("pixelPIN", pixelToggle);
}
  void loop (){

   if(toggle==true){

  colorWipe(strip.Color(0, 0, 255), 25); // Blue
  colorWipe(strip.Color(0, 0, 255), 25); // Blue
  colorWipe(strip.Color(150, 150, 150), 25); // White
  colorWipe(strip.Color(150, 150, 150), 25); // White
  colorWipe(strip.Color(255, 0, 0), 25); // Red
  colorWipe(strip.Color(255, 0, 0), 25); // Red
  strip.setPixelColor(0,0,0,255);
  strip.setPixelColor(3,0,0,255);
  strip.setPixelColor(6,0,0,255);
  strip.setPixelColor(9,0,0,255);
  strip.setPixelColor(12,0,0,255);
  strip.setPixelColor(15,0,0,255);
  strip.setPixelColor(18,0,0,255);
  strip.setPixelColor(21,0,0,255);
  strip.setPixelColor(1,150,150,150);
  strip.setPixelColor(4,150,150,150);
  strip.setPixelColor(7,150,150,150);
  strip.setPixelColor(10,150,150,150);
  strip.setPixelColor(13,150,150,150);
  strip.setPixelColor(16,150,150,150);
  strip.setPixelColor(19,150,150,150);
  strip.setPixelColor(22,150,150,150);
  strip.setPixelColor(2,255, 0, 0),
  strip.setPixelColor(5,255, 0, 0),
  strip.setPixelColor(8,255, 0, 0),
  strip.setPixelColor(11,255, 0, 0),
  strip.setPixelColor(14,255, 0, 0),
  strip.setPixelColor(17,255, 0, 0),
  strip.setPixelColor(20,255, 0, 0),
  strip.setPixelColor(23,255, 0, 0),
  strip.show();
  delay(5000);
  colorWipe(strip.Color(0, 0, 0), 0); // off
  (toggle = false);
   }

    else {

    comet();
    }
  }

   void comet (){
        for ( int i=0 ; i &lt; PIXEL_COUNT * 5 ; i++ ) {
        for ( int p=0 ; p &lt; PIXEL_COUNT ; p++ ) {
            strip.setPixelColor( p, 0 );  //  clear the strip
        }
        drawFade( i, 255, 255, 149);        //  then draw the fade
        strip.show();
        delay( 400 );
    }
}

void drawFade ( uint8_t start, uint8_t r, uint8_t g, uint8_t b ) {
    int brightness = 50;
    for ( int p=0 ; p &lt; FADE_LENGTH ; p++ ) {
        strip.setPixelColor(
            ( start - p ) % PIXEL_COUNT,
            strip.Color(
                ( r * brightness ) / 255,
                ( g * brightness ) / 255,
                ( b * brightness ) / 255
            )
        );
        brightness = ( brightness * FADE_SCALE ) / 255;
    }
} void colorWipe(uint32_t c, uint8_t wait) {
  for(uint16_t i=0; i&lt;strip.numPixels(); i++) {
    strip.setPixelColor(i, c);
    strip.show();
    delay(50);
  }
}

int pixelToggle(String command){
    toggle = true;
}
```

Set Up IFTTT

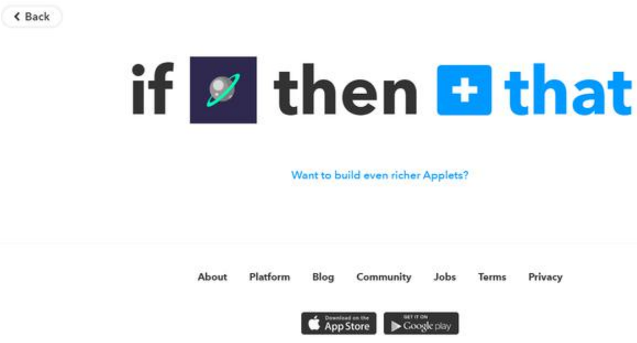
Make sure your Photon is still plugged into your computer for this next step.

Head over to IFTTT (IfThisThenThat) () and set up a sign-in. IFTTT is an Internet-of-Things site that allows you to connect different devices using easy-to-use applets. Not only can you connect Hue lights to trigger certain colors based on weather conditions, you can also connect microcontrollers that have Wi-Fi capability. With your account now set, hover over your account name in the right-hand corner and choose New Applet from the pull-down menu.



IFTTT wants you to think literally about two things: "this" happens triggering "that". So, we will choose these two things in different steps. Here goes:

1. Click on the +this and type space in the search bar
2. Click on the Space square that appears
3. Now select the square ISS passes over a specific location
4. Move the map to select your exact location and click Create Trigger



OK, now we are shifting gears to handle the second part.

1. Click on the +that and type Particle in the search bar
2. Click on the Particle square that appears

3. Now select the square Call a function
4. Under Then Call (Function Name) (required), check the dropdown for pixelPIN on "yourdevicename"
5. Now go to the bottom and click Complete Action
6. You can keep the green light that enables Receive notifications when applet runs
7. Hit Finish
8. You can Check Now to see that the applet is working, but it will only do the special light sequence if you force the trigger or the ISS actually passes
9. If you ever need to check this applet, you'll find it under My Applets on the top menu
10. Be aware that if you continue to run the applet without your pin having power, it will eventually automatically shut down
11. It's nice to have the IFTTT app installed on your phone to turn applets on/off

Want a Test?

If you need to test what a flyby would be like without waiting long hours, it is good to set up another applet. This one will allow you to send an email to trigger your pin.

1. Follow the above directions for creating a new applet
2. This time for +this type Email in the search bar
3. Choose the Send IFTTT any email box
4. Now follow the directions above for doing the +that choosing Particle
5. Keep following those directions for all the choices including the function
6. Under the function section you'll see +ingredient - select subject
7. Hit Create Action and then Finish

This applet will enable you to send an email to trigger@applet.ifttt.com with anything written in the subject line. In a few seconds the pin will do its special light show as if the ISS had flown by. This is very handy for demos.

Wear It!

The best way to wear the pin is to plug in the USB battery pack and tuck it in your pocket. It can last 24 hrs. even with all the pinging of data. If you want to go rogue you can plug in the little LiPoly battery (the plug placement is a bit awkward and will take some finger gymnastics). You can trim a small piece of velcro and attach it to the battery with its matching velcro side on the back of the pin. That way you can just wear the battery for meet-and-greets for a short amount of time. Remember to use the micro Lipo charger for the battery. Also, if you want to change the Wi-Fi zone of your pin, you will need to review the basics (). You may also want to install Particle's Command Line () to simplify things.

If you are a serious ISS fan, you will probably want to sign up for alerts from NASA () for best viewing days/times. This pin's data can be up to an hour off according to Particle, but it does let you know all the passes for your neighborhood, which is cool. The idea for the pin stems from a lack of awareness of how frequently the ISS is flying overhead. So, think of this project less as a scientific device, but more of a mindful way of experiencing spaceflight. Sending high fives to my NASA Datanaut friends!