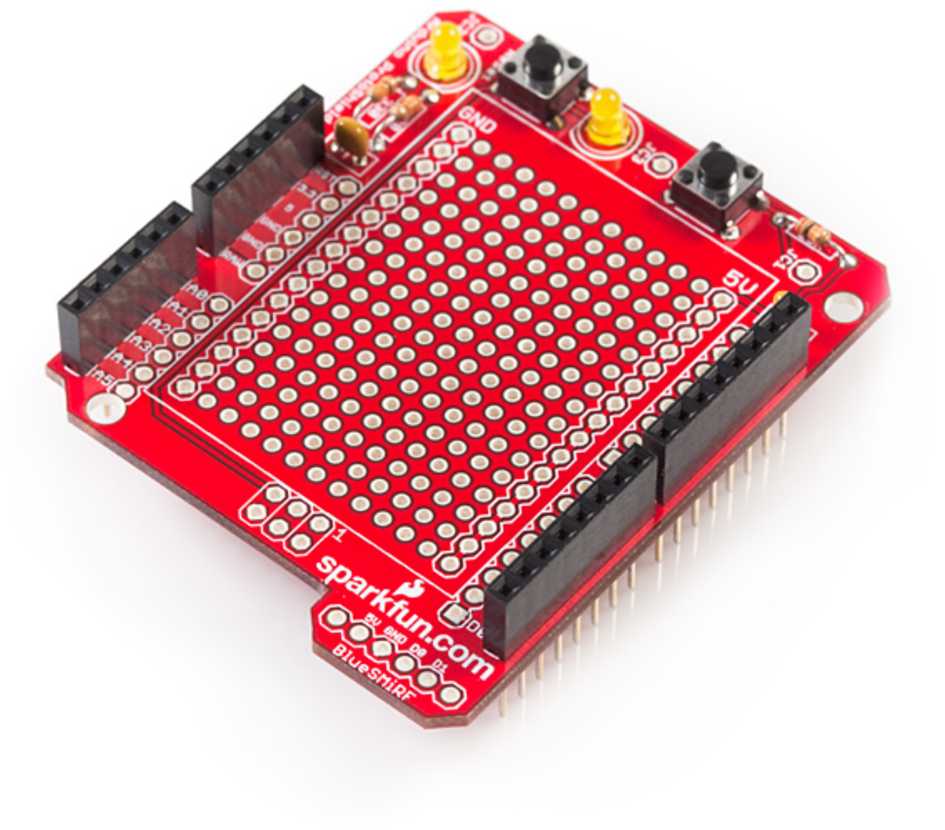
Home | Tutorials | ProtoShield Quickstart Guide

ProtoShield Quickstart Guide
by Jimb0 | June 24, 2010 | *2 comments* Skill Level: ☆ Beginner

Arduino ProtoShield

You've got a project idea, you have an Arduino, now what you really need is a platform to actually build your new project onto. You need something that will easily connect your Arduino to all the external bits and pieces your project requires. Enter, the Arduino ProtoShield.



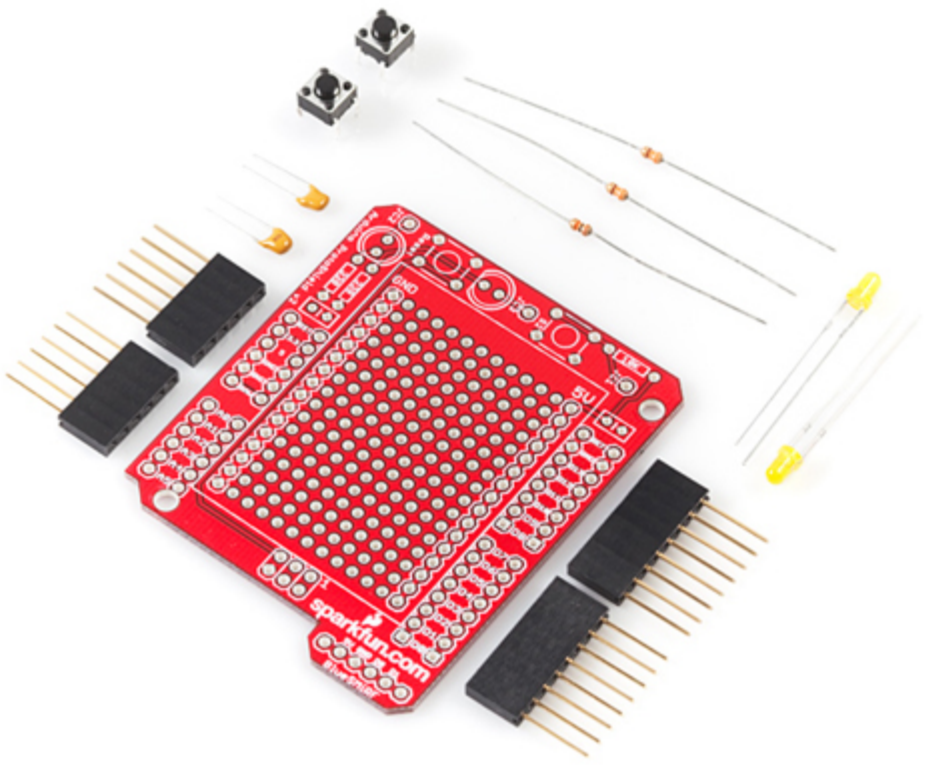
What is the ProtoShield?

The ProtoShield is designed to **facilitate prototyping** - it allows for easy connections between a breadboard and an Arduino. The ProtoShield provides your Arduino with easy access to a **prototyping area**, two general use **LEDs**, access to a BlueSMiRF socket, a general use **push button**, and - most important of all - the Arduino **reset button**. There's also an option of sticking a mini, self-adhesive breadboard onto the ProtoShield - keeping it reusable for all of your future projects.

The ProtoShield is one of many Arduino shields. Shields are designed to sit on top of an Arduino, extending the functionality of the popular microprocessor platform.

How do I assemble it?

The ProtoShield is shipped in kit form - as a box of parts and a PCB. We've written a step-by-step assembly tutorial to guide you in building your new shield. Don't worry if you've never soldered before, this is a great kit for beginners!



How do I use it?

Well, that's mostly up to you! There are a few things I will point out, though:

Using the LEDs
As the assembly guide points out, you have to wire the LEDs to whichever Arduino pin you'd like. Once you've done that, you can program your Arduino to run the ever-important 'Blink' sketch (in Arduino, go to File->Examples->Digital->Blink):

```
int ledPin =  13;

void setup()   {
  pinMode(ledPin, OUTPUT);
}
```

```
void loop()
{
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```

Make sure you change the value of '**ledPin**' to whatever digital pin your LED is connected to; in the above example it is tied to pin 13. That sketch should turn the LED on for a second, off for a second, on for a second, off for a second, and so on.

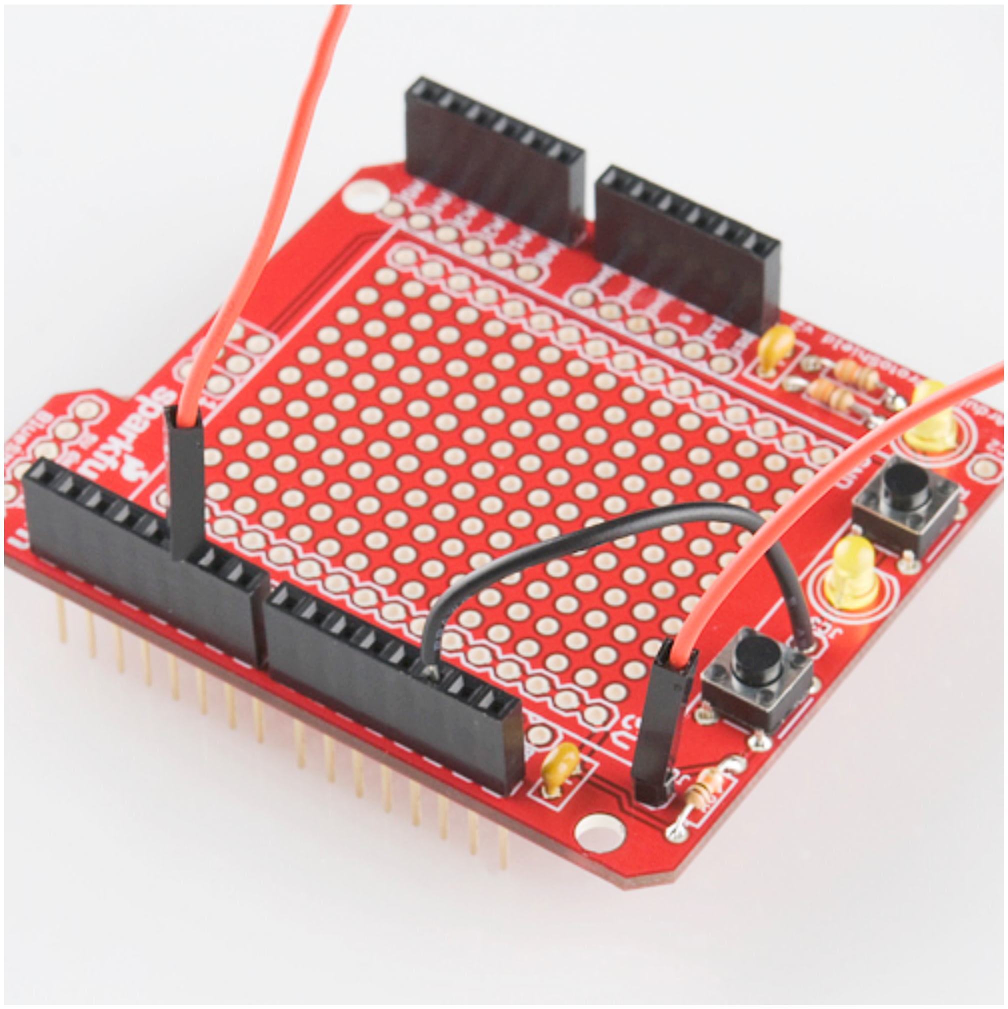
Note that both LEDs have one leg tied to ground through a 330Ω resistor, and the other leg connected to either JC2 or JC3 (and then to whatever pin you have them connected to on your Arduino). This means that you have to '**digitalWrite**' the pin **HIGH** to turn the **LED on**, and **LOW** to turn it **off**.

Sidenote: The 330Ω resistor is there to limit the current that is allowed to go through the LED. This is a necessity for just about every LED, as they are prone to blow up if supplied current that is over their maximum rating (about 20mA for these LEDs).

Using the Button
As with the LEDs, before you can use the button you have to connect it to a digital or analog pin of your choice. For a perfect example of how to use the button, check out the 'Button' example (in Arduino go to File->Examples->Digital->Button):

```
const int buttonPin = 2;
const int ledPin =  13;

int buttonState = 0;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}

void loop(){
  buttonState = digitalRead(buttonPin);

  if (buttonState == HIGH) {
    digitalWrite(ledPin, HIGH);
  }
  else {
    digitalWrite(ledPin, LOW);
  }
}
```

Make sure you change '**buttonPin**' and '**ledPin**' according to how you've wired them, the example assumes they're connected to digital pins 2 and 13, respectively.

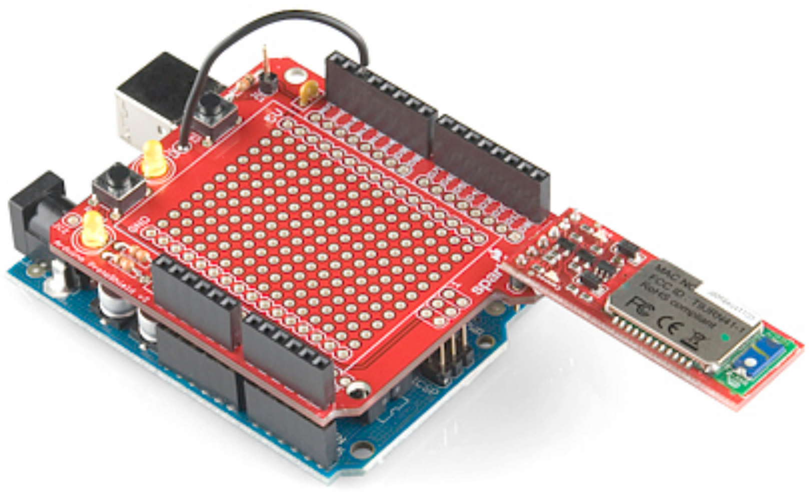


The **button** on the ProtoShield is '**pulled high**', meaning one side of the button is connected, through a resistor, to +5V and also to 'JC1' (which should then be connected to an Arduino pin of your choice). The other end of the button is simply connected to ground. This means that when the **button is pressed** it will **read as LOW**, otherwise it will be HIGH. In the example above, the LED will be on unless the button is being pressed.

Using the BlueSMiRF port
Bluetooth is a really simple, and effective way to add wireless communication to your project. The BlueSMiRF Gold works as, essentially, a serial cable minus the wires. Only four connections are required to communicate with the BlueSMiRF - 5V, GND, RX and TX.

When you connect the BlueSMiRF to your ProtoShield, take care to match the

'VCC', 'GND', 'TX-O', and 'RX-I' labels of the BlueSMiRF with the '5V', 'GND', 'D0', and 'D1' labels of the shield.



You can use the standard 'Serial.print' and 'Serial.read' commands to send and receive serial data. Here's an example that builds on the previous code, adding serial communication:

```
int ledPin = 13;
int buttonPin = 2;

int buttonState = LOW;
int oldButtonState = HIGH;
char serInput;

void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
  digitalWrite(ledPin, HIGH);

  Serial.begin(115200);
}

void loop()
{
  buttonState = digitalRead(buttonPin);

  if (Serial.available() > 0)
  {
    serInput = Serial.read();
    Serial.println("Miny, Moe.");
  }
  if (buttonState != oldButtonState)
  {
    if (buttonState == HIGH)
    {
      digitalWrite(ledPin, HIGH);
      Serial.print("Meeny, ");
    }
    else
    {
      digitalWrite(ledPin, LOW);
      Serial.print("Eeny, ");
    }
  }

  oldButtonState = buttonState;
}
```

This example will actually work with both the standard USB serial connection, or a BlueSMiRF. Give it a try! Note that the baud rate is set to **115200 bps**, this is the **default** baud rate of the BlueSMiRF. Pressing the button will turn the LED off and send "Eeny, " out the serial port. Releasing the button will turn the LED back on and print "Meeny, ". Sending any character to the Arduino, over a serial connection, will print "Miny, Moe.".

If you want to communicate between your computer and the BlueSMiRF, you will need a bluetooth dongle of some sort. Many options exist to add bluetooth to your computer (if it doesn't already have it). For beginners, I'd recommend the Bluetooth USB module; another, more advanced, option is the USB BlueDongle.

How have others used the shield?

The ProtoShield can be used to prototype just about any Arduino project you could imagine. As examples of the huge variety of projects the shield has been used for, check out these links:

- Arduino USB Host
- Ultrasonic Range Finder Tape measure

- Joystick Control of Servo Motors
- 4-Bit Maze
- Temperature Monitor/Controller
- Weather Station Receiver
- Basic 555 Timer

Do you have an example you'd like posted here? Drop us an email!

Schematic and PCB Layout

For technical information including the schematic and design files, please see the product page. You may also want to add/review the comments on that page or do a Google search for more example projects that use the ProtoShield.

Have a suggestion for how we can improve this quickstart guide? Concepts not explained clearly? Need more example code? Please let us know. You can leave a comment below or email us spark@sparkfun.com. Also let us know if this is the most awesome Quickstart guide you have ever encountered and we will stop trying to improve it.

▼ Comments 2 comments

Login to post comments.

- ldattilo | about 10 months ago 1

  What connector did you use in the picture to connect the BlueSMiRF?

  - Eric Falsken | about 3 months ago 1

    The BlueSMiRF is wired to the 6 pin holes along one side that are labeled for such. Tx and Rx are wired to D0 and D1 for you.