LoRa Remote Relay

We are very excited to get the chance to play with our new LoRa Shields, and test just how far we could get a signal using LoRa technology. With one of our tests, we were able to get a signal to travel about 200m, including through the corrugated iron of the warehouse and the concrete office building. We set about building a practical project that would make best use of the technology. The LoRa Remote Relay is basic- four buttons at one end and four relays at the other end. It also has LED's next to the buttons, so unlike other remote relay systems, there is feedback that the relays are operating as commanded.

Shopping List:

2 x XC4410 Uno Main Board (Leonardo or Mega boards will both work fine)
2 x XC4392 LoRa Shield
1 x XC4482 Prototyping Shield
4 x SP0601 Tactile Pushbutton Switch
1 x RR0564 Pack of 470 Ohm resistors
4 x ZD0250 Red/Green Bicolour LED's
1 x XC4440 Four Channel Relay Module
1 x WC6028 Plug-Socket Jumper Lead Set

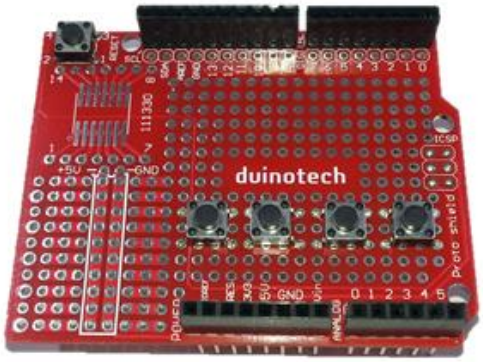
Construction:

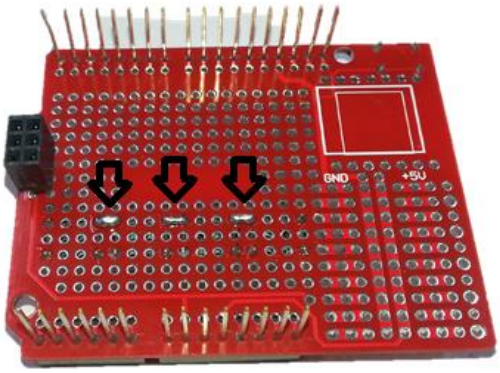
A good first step is to prepare both LoRa Shields. Attach the antenna, and ensure that the jumper shunts on J_DIO5, J_DIO2 and J_DIO1 are removed. We've just moved them to one side so they aren't bridging the jumper connection. Check the position of the other jumpers too.

Transmitter Unit

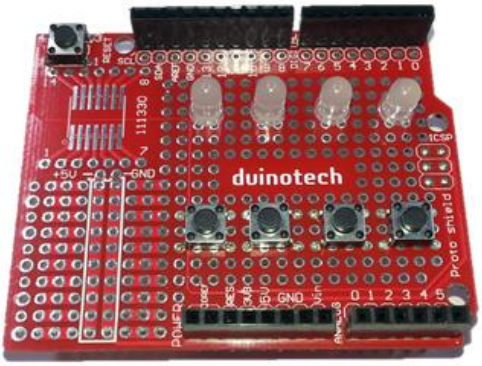
The transmitter unit is the more involved of the two, but all the soldering is done on the Prototyping Shield to make it easier to reuse the parts.

Start by placing the Tactile Pushbuttons onto the shield, and soldering in place. Note the solder bridges along one edge- this simplifies our wiring, as the switches are connected internally.



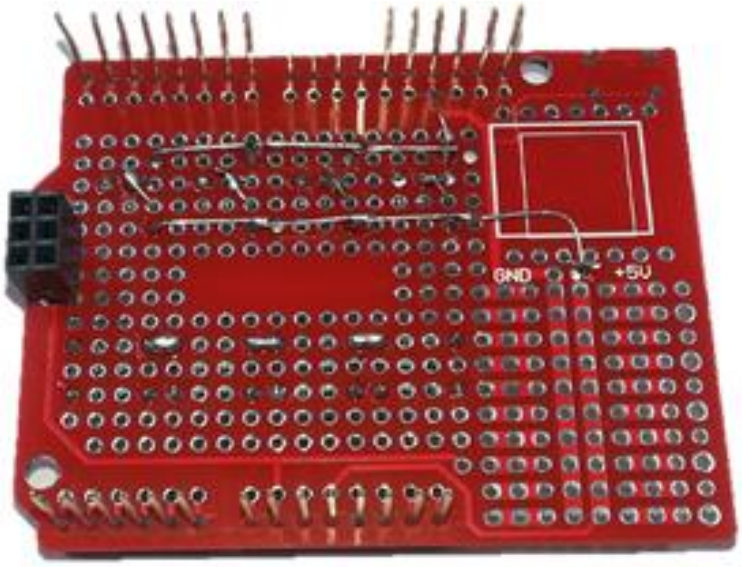


Next, install the LED's, and solder in place. Note that the flats on the LED's is on the right, closest to the ICSP header. If the LED's are installed backwards, they will light up the wrong colour.



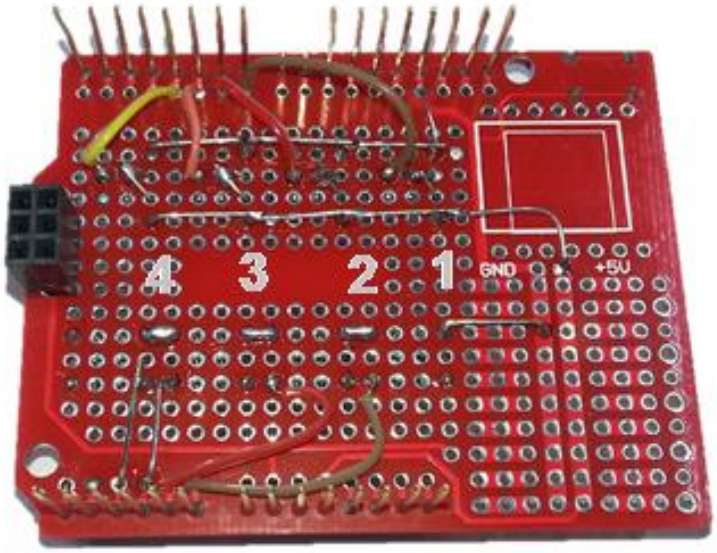
The resistors are installed next. We found it easier to bend all the legs around first, then place the resistors. You can solder the resistors in place, but don't trim the legs yet, as we're going to use them to make some of the connections on the Prototyping Shield.

Note that the inner resistor legs are soldered onto the closest LED leg (which should be the side without the flat), and the outer legs of the resistors are run into each other to form a bus.



The resistor legs should be bent to the right to adjoin the next resistor, and the last leg taken to either the GND connection (top row) or the 5V connection (bottom row).

The final step is to take some offcuts of the WC6028's and some resistor legs (these are easier to handle as they don't have to be stripped, but you do have to be careful to make sure they don't short-circuit anything), and make the following wiring connections.



ProtoShield Pin	Connection
A0	Button 1
A1	Button 2
A2	Button 3
A3	Button 4
D7	LED 1
D6	LED 2
D5	LED 3
D4	LED 4

Note also that the bus we created earlier by bridging the pushbuttons is extended to the GND rail on the right of the Prototyping Shield. The wiring for the LED's might look a bit complicated, but it allows us to control each LED with just a single IO pin. The above pin allocations are also set by #defines in the sketch, so they can also be changed if you need to alter what pins are used.

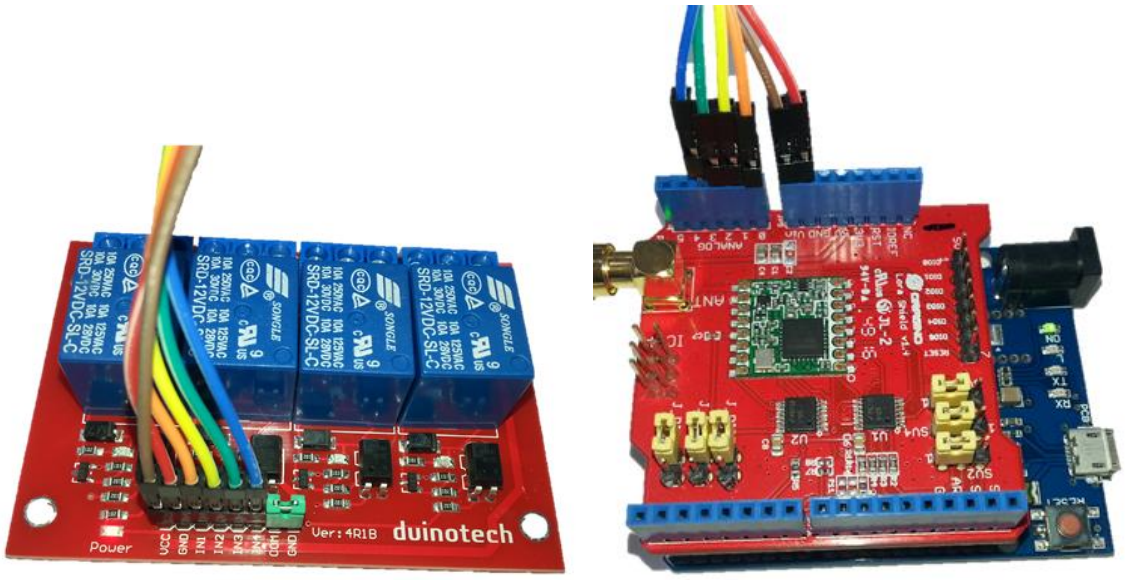
To complete assembly of the transmitter unit, attach the LoRa Shield to the Uno, then attach the assembled Prototyping Shield onto the LoRa Shield.

Receiver Unit

There is no soldering needed for the receiver unit, just a set of six jumper leads. Plug the other LoRa Shield into the top of the other Uno, then make the connections between the LoRa Shield and the Relay Module as shown below. As for the transmitter, the below pin connections can be changed in the sketch.

LoRa Shield Pin	Relay Module	Colour
GND	GND	Brown
VIN	VCC	Red
A0	IN1	Orange

A1	IN2	Yellow
A2	IN3	Green
A3	IN4	Blue



The relays require 12V to operate, so when the LoRa Remote Relay is put in use, the receiver needs to be fed from a 12V supply through the DC jack of the Uno Board. You can still do testing from USB power, as the LED's on the relay board will light up, even with just 5V.

Code:

Apart from the inbuilt SPI library, there is one external library for the LoRa Shields. This can be installed via the Library Manager (Sketch>Include Library>Manage Libraries…). Search for 'lora', and install the library simply called 'lora'. Alternatively, it can be downloaded from https://github.com/sandeepmistry/arduino-LoRa.

Open the LoRa_Remote_RX sketch, select the correct main board and COM port, then upload to the receiver unit. Do the same with the LoRa_Remote_TX for the transmitter unit. There are no defaults that need to be changed in the sketch if the pins have been wired as above.

Testing and Use:

The basic operation involves pressing the button on the transmitter unit to toggle the status of the relay on the receiver unit. The LED colour indicates the currently selected relay state (green for ON, red for OFF). If the LED is solid, then the transmitter has received feedback that the state is correct. If the LED is flashing, the transmitter has not received correct feedback.

With both units powered up, the LED's should be flashing red on the transmitter unit, and none of the relay LED's should be on. Wait about 20 seconds, and the LED's on the transmitter should go to solid red (this indicates that they are getting feedback from the receiver).

Press one of the buttons on the transmitter unit, and the corresponding LED should turn green. This indicates that it is trying to turn the relay on. If the link is working, the corresponding relay should turn on, and the LED will be solid green. If the relay doesn't turn on, then the link from the transmitter to receiver might not be working. If the LED stays flashing for more than 20 seconds, the link from the receiver to transmitter might not be working.

As well as transmitting status back and forth when a button is pressed, the signals are also resent every 10 seconds or so. You can see this effect by pressing multiple buttons at the same time, as the transmitter can only send one signal at a time, and will eventually update all relays.

Improvements:

The success of a project like this is using it in a real world application. We've heard of a few people using remote relays to control motorised gates. The advantage of the LoRa Remote Relay in this case is that the actual position of the gate could also be monitored and fed back to the transmitter unit, so that you could not only know that the motors are working, but that the gate has reached its endpoint. Or you could use it for remote pump operation, and also feed the data about tanks levels back to the user.

The sketches use the basic default settings for the shield. You can experiment with some of the settings inside the library. These are documented here: https://github.com/sandeepmistry/arduino-LoRa/blob/master/API.md. You may need to make sure that both transmitter and receiver use the same parameters, otherwise they may not 'hear' each other. The testing we did was in an urban area around concrete and corrugated iron, and longer ranges should be achievable with clear line of sight. Increasing the spreading factor and coding rate may also improve signal robustness at the expense of transmission speed.

Of course, this is just one use of the LoRa Shield- many systems that work with serial data can be modified to work with LoRa- the data just needs to be assembled into packets to be sent. You could even have multiple sensors in the field feeding data back to a central receiver.

Sketch:

The sketch consists of two folders, one for the transmitter unit and one for the receiver unit. They are available for download in the attached zip file.