BIRDS ON THE WIRE

Birds on the Wire is the Little Bird Electronics blog.

Tutorial: Arduino and Infra-red control

Welcome back fellow Arduidans!

In this article we will look at something different to the usual, and hopefully very interesting and useful – interfacing our Arduino systems with infra-red receivers. Why would we want to do this? To have another method to control our Ardiuno-based systems, using simple infra-red remote controls.

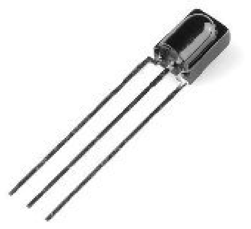
A goal of this article is to make things as easy as possible, so we will not look into the base detail of how things *work* - instead we will examine how to get things *done*. If you would like a full explanation of infra-red, perhaps see the page on **Wikipedia**. The remote controls you use for televisions and so on transmit infra-red beam which is turned on and off at a very high speed – usually 38 kHz, to create bits of serial data which are then interpreted by the receiving unit. As the wavelength of infra-red light is too high for human eyes, we cannot see it. However using a digital camera – we can. Here is a demonstration video of IR codes being sent via a particularly fun kit – the adafruit ***TV-B-Gone****:*

Now to get started. You will need a remote control, and a matching IR receiver device. You can purchase a matching set for a good price,**such as this example**:

Or you may already have a spare remote laying around somewhere. I kept this example from my old Sony Trinitron CRT TV after it passed away:

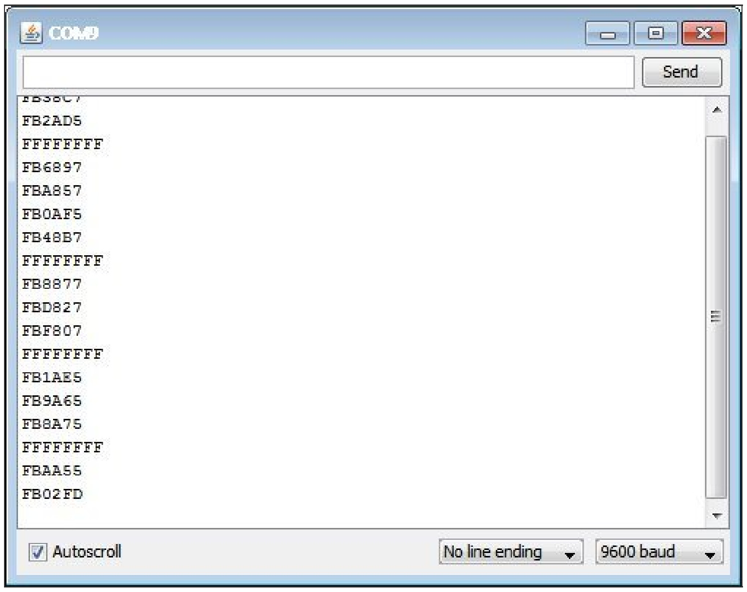
It will more than suffice for a test remote. Now for a receiver – if you have purchased the **remote/receiver set**, you have a nice unit that is ready to be wired into your Arduino, and also a great remote that is compact and easy to carry about. To connect your receiver module – as per the PCB labels, connect Vcc to Arduino 5V, GND to Arduino GND, and D (the data line) to Arduino digital pin 11.

Our examples use pin 11, however you can alter that later on. If you are using your own remote control, you will just need a receiver module. These are very cheap, and an ideal unit is the Vishay TSOP382. They are also dead-simple to use. Looking at the following example:



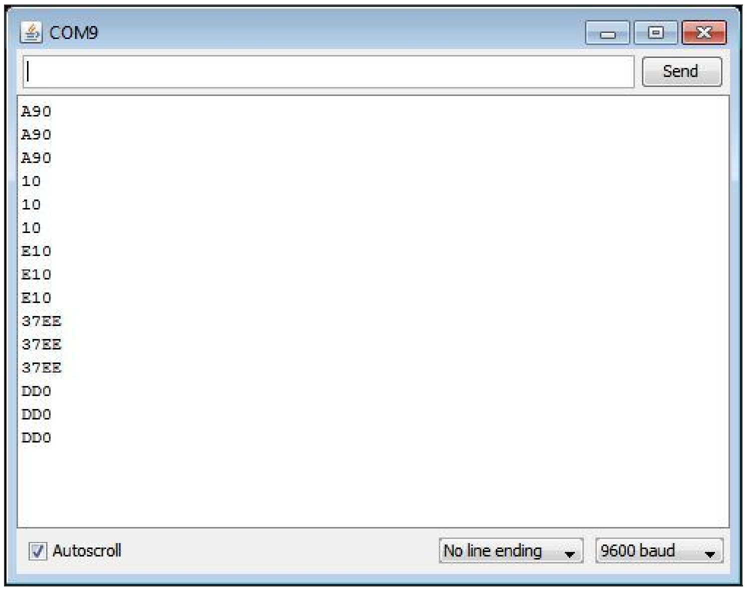
From left to right the pins are data, GND and Vcc (to Arduino +5V). So it can be easily wired into a small breadboard for testing purposes. Once you have your remote and receiver module connected, you need to take care of the software side of things. There is a new library to download and install, **download it from here**. Extract the *IRremote* folder and place into the *..\arduino-002x\libraries* folder. Then restart your Arduino IDE if it was already open.

With our first example, we will receive the commands from our remote control and display them on the serial monitor. **Download** and run the this sketch (example 32.1). Open the serial monitor box, point your remote control to the receiver and start pressing away. You should see something like this:



```
FB2AD5
FFFFFFFF
FB6897
FBA857
FB0AF5
FB48B7
FFFFFFFF
FB8877
FBD827
FBF807
FFFFFFFF
FB1AE5
FB9A65
FB8A75
FFFFFFFF
FBAA55
FB02FD
```

What have we here? Lots of hexadecimal numbers. Did you notice that each button on your remote control resulted in an individual hexadecimal number? I hope so. The number FFFFFFFF means that the button was held down. The remote used was from a yum-cha discount TV. Now I will try again with the Sony remote:



```
A90
A90
A90
10
10
10
E10
E10
E10
37EE
37EE
37EE
DD0
DD0
DD0
```

This time, each button press resulted in the same code three times. This is peculiar to Sony IR systems. However nothing to worry about. Looking back at the sketch for example 32.1, the

```
if (irrecv.decode(&results))))
```

section is critical – if a code has been received, the code within the *if* statement is executed. The hexadecimal code is stored in the variable

```
results.value
```

with which we can treat as any normal hexadecimal number. At this point, press a few buttons on your remote control, and take a note of the matching hexadecimal codes that relate to each button. We will need these codes for the next example…

Now we know how to convert the infra-red magic into numbers, we can create sketches to have our Arduino act on particular commands. As the IR library returns hexadecimal numbers, we can use simple decision functions to take action. In the following example, we use*switch…case* to examine each inbound code, then execute a function. In this case we have an **LCD module connected via I2C**, and the sketch is programmed to understand fifteen Sony IR codes. If you don't have an LCD you could always send the output to the serial monitor.

Furthermore you can substitute your own values if not using Sony remote controls. Finally, this sketch has a short loop after the*translateIR();* function call which ignores the following two codes – we do this as Sony remotes send the same code three times. Again. you can remove this if necessary. Note that when using hexadecimal numbers in our sketch we preced them with *0x*. Download the sketch example 32.2 **from here**. Now here it is in action:

You might be thinking "why would I want to make things appear on the LCD like that?". The purpose of the example is to show how to react to various IR commands. You can replace the LCD display functions with other functions of your choosing.

At the start working with infra-red may have seemed to be complex, but with the previous two examples it should be quite simple by now. So there you have it, another useful way to control our Arduino systems. Hopefully you have some ideas on how to make use of this technology. In future articles we will examine creating and sending IR codes from our Arduino. So stay tuned for upcoming Arduino tutorials by subscribing to this blog, following us on twitter or facebook. Furthermore, a big thanks to **Ken Shirriff** for his **Arduino library**.

Otherwise, have fun, stay safe, be good to each other – and make something!

Posted 1 year ago by John Boxall | Viewed 12349 times | Favorited 0 times

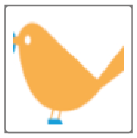
Tweet 2 Like 1

0 Comments

Leave a comment...

To leave a comment on this posterous, please login by clicking one of the following.

posterous login or signup login with twitter



littlebird

littlebirdelectronics.com

FriendfeedFlickrYoutubeViddlerVimeoDelicious

Subscribe...

Subscribe via RSS

Follow by email »
Get the latest updates in your email box automatically.

Tags

arduino (11)
tutorial (7)
electronics (6)
bird (5)
little (5)
tutorials (4)
arduino tutorial (3)
example (3)
diy (3)
tinygps (2)
View all 184 tags ↓

Contributors

John Boxall

littlebird

This blog is by the **Little Bird Electronics** team.