# DIY Headtracker with Bluetooth – ver 1.06

Chris Stephens (RCGroups user : Hangarspace) – October 2018

Starting with version 1.04 of Denis’ software, I have extended it to add support for a Bluetooth connection between the headtracker unit and the RC transmitter. This new version still supports the original connection-by-cable option.

The headtracker end uses the same Arduino and sensor board as the original project, but it also uses an HC06 Bluetooth board. The RC transmitter end uses another Arduino Nano and an HC05 Bluetooth board.

The trickiest part of the project is configuring the HC05 to automatically connect to the HC06. Boards that are a few years old will most likely have a version 2.x firmware which is well documented, but newer ones have a version 3.x firmware for which the pairing process differs and is poorly documented.

I bought over a dozen different boards from different sources to try to find a source of boards running firmware version 2.x but with no luck. This put me off publishing the project as I couldn't provide a clear set of steps on how to pair the version 3 Bluetooth boards and I'm not around very much to provide close support.

However, over the summer of 2018 somebody worked out the steps and was kind enough to publish them (link below).

Whilst following those steps I found that most HC05's will pair with any HC06 that they find in range without any other configuration, so maybe so it only becomes a potential problem if you have multiple HC06's in range. Even then, you might be able to convince them to pair by setting the password to something that doesn't match anything else in range, obviously you must make sure that you set the same password on the HC05 and HC06 that you intend to use.

The next gotcha is in the choice of pins we use to connect the Bluetooth boards to the Arduino. The Arduino Nano has a dedicated serial port controller that provides two ‘hardware’ serial port pins. The catch is that this same port is used by the USB connection and so we need to provide a way to disable the Bluetooth module so that we can use the USB connection. Arduinos do allow for emulated serial ports to run on other pins, but tests have shown that the processing overhead of using these results in very poor performance and garbled data.

To provide power to the Bluetooth boards, I'm taking a parasitic approach by tying the boards VCC to a pair of I/O pins set to be 'outputs'. It's important to use two pins to share the current load. I opted to use pins D11 and D12, which meant moving the switch to D10. Powering them this way also allows us to turn them 'off' in code which helps when we want to use the USB cable. Technically the rx and tx levels on the BT boards are only 3.3v so the Arduino will be overdriving them, but I haven't found this to be a problem yet.

To disable the Bluetooth operation, power up the device whilst holding the button connected to pin D10. You need to do this on the module that you want to disable BT for, so each module needs a button.

I have also included the ability to have a button on the RC transmitter end which will pass thru any ‘presses, to perform the 're-centre reset'. Pressing the button on the RC transmitter module will NOT disable the BT module.

It still supports the ability to ‘pause’ the head tracker using a brief press of the button (at either end), but this will NOT disable the Bluetooth.

The process for loading the firmware onto the Arduinos is the same as the original project using the Arduino IDE. Because of the complication around sharing the serial port, you will find it easier to do the flashing, configuration and calibration with the headtracker board set to the original 'Standard' mode first, all the way thru to wiring it up to your trainer port and making sure your pan and tilt works before connecting the Bluetooth HC06 board and changing its mode to 'BT Headtracker'.

Using the new version of the GUI, there is an additional option that allows you to set the device to run in either the original mode (Standard) or as either a Bluetooth enabled headtracker or receiver. Once you have configured the headtracker module, then save the same configuration to the receiver module, but don't forget to set its device mode to 'BT Receiver'.

When it comes to mounting the receiver modules, my first approach was to fit them inside an external module box that allowed me to fit it into the back of my Taranis X7, take power from the module bay pins and have a short cable connection into the trainer port. Then, in the model settings I enable the 'external rf' which turns on the power to the external module pins and setup the trainer options accordingly. I actually created two flight modes, one where the pan/tilt is controlled by the pots and then a second mode which is activated by a switch that uses the trainer port for pan/tilt.

This worked great, until I recently purchased an R9M module. To support both items, I relocated the headtracker receiver modules inside the transmitter case. It was a very tight fit and I had to bend/trim a few pins off to ensure it didn't snag with the gimbal.

Sorry it's a bit wordy, but hopefully you will find it useful info. Like I mentioned at the top, the trickiest bit was configuring the Bluetooth modules to pair, which in my case since I have multiple HC06's involved elsewhere in my fpv setup was more important. You should be able to figure it all out even if you need to exercise your brain a little. If nothing else you will hopefully learn a bit along the way.

# Steps to Success

Bluetooth Pairing

You will need a USB FTDI device to allow you to connect the Bluetooth boards to your PC for configuration. These devices are widely documented on the internet, so to get a basic understanding of them, Google is your friend. Suffice to say that the HC06 acts as a slave and the HC05 acts as the master. The HC05 is the device which searches for an HC06 to pair with.

1. Follow online guides for setting the BAUD rate to 57600 on both devices.

Version 2 HC06 : AT+BAUD8

Everything else : AT+UART=57600,0,0

1. Optionally, change the standard password on both devices.
2. Power up both HC05 and HC06 and see if they pair by default. You can use the LED indicators to see if they have paired. Quick repeating flashes indicate they are searching, a long off-pause with two quick flashes indicate they have paired, hopefully with each other!  
   If they pair up and you are happy with that then move on to the construction of the Arduino modules. If not then follow the steps on this link to configure the pairing.  
     
   <https://arduino.stackexchange.com/questions/50974/how-to-solve-problem-atinq-error-1f-atinit-error17-on-bluetooth-module-hc>   
     
   At the bottom of that page, is a post by Presidenten who was able to pair his version 3 devices.

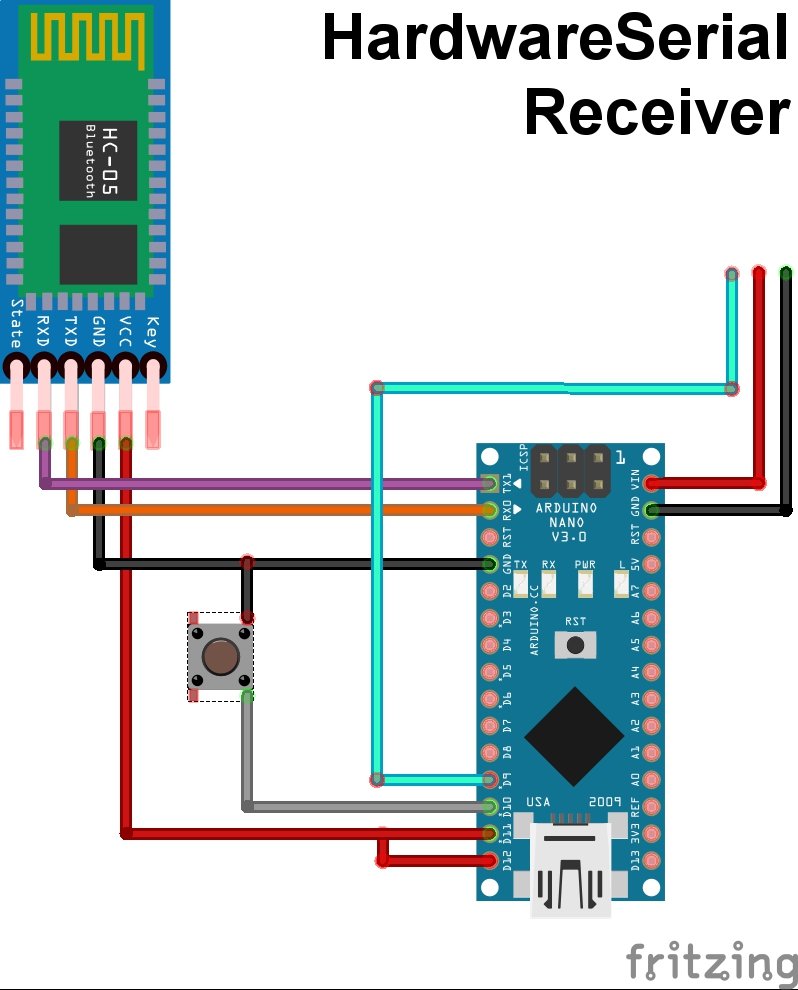
Headtracker Module Construction

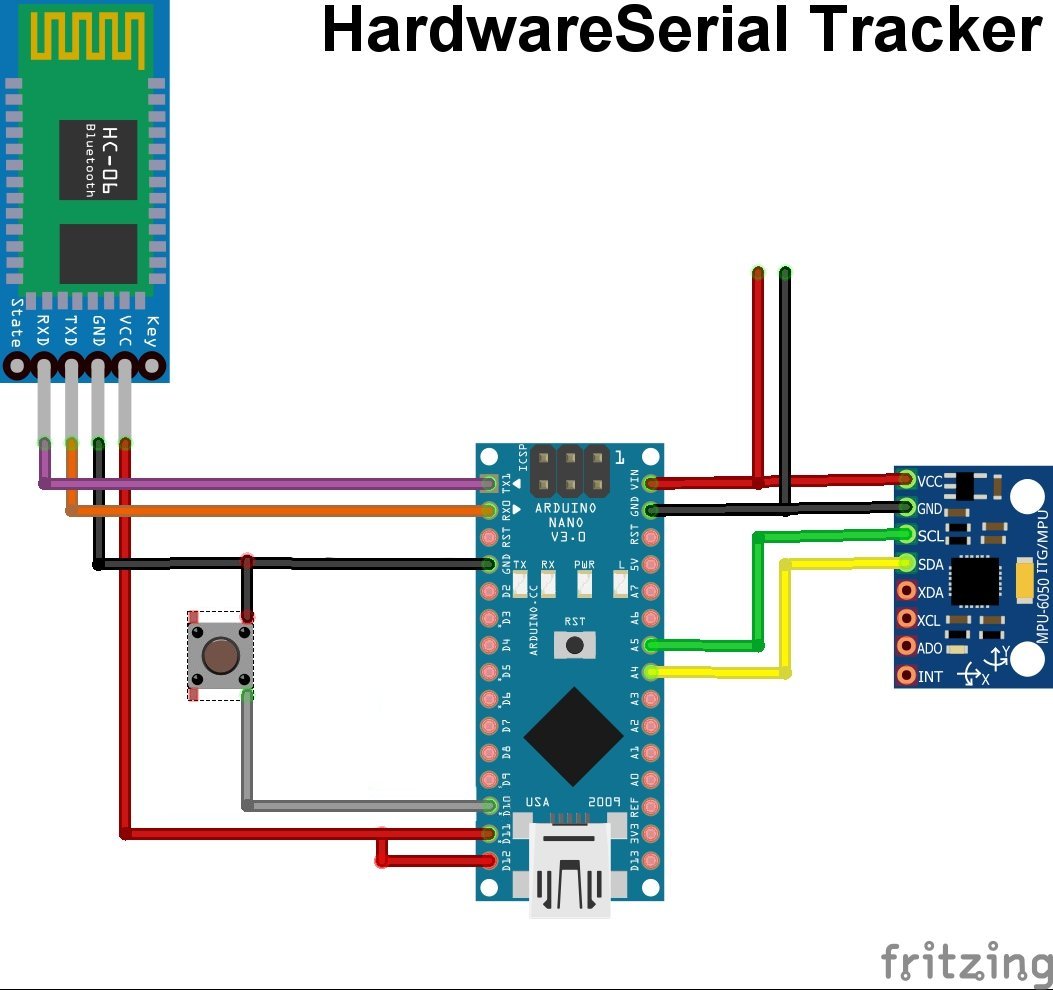
1. Construct the head tracker module as per the original project with the exception of connecting the momentary button to pin D10
2. Load the firmware onto the module using the Arduino IDE
3. Use the HeadtrackerGUI program to calibrate and configure the module. It’s advisable to set the ‘device mode’ to standard and connect it up to your RC transmitters trainer port from D9 and check it is all working as expected.
4. Once you are happy with the configuration, change the device mode to ‘BT Tracker’, save the settings to the device and also save the settings to file as we will use them later.
5. Connect the HC06 Bluetooth module as shown in the diagram.

RC Transmitter Module Construction

1. Load the firmware onto the Arduino using the IDE
2. Connect the Arduino to the GUI app, load up the saved settings file, change the device mode to ‘BT Receiver’ and then save the settings to the device.
3. Connect up the HC05 Bluetooth module as per the diagram.
4. Wire the module into your radio in such a way that you supply it with power and can connect to the trainer port.

Go fly!





## Technical Notes

If you want to know how the data is sent, then the approach I took was to interrupt the process so where it would previously have passed the new servo position values to the PPM generation routines, it now just sends those values over the serial connection which then get picked up by the receiver module and passed into its PPM generation routine that then provides the output signal for your trainer port.

For efficiency, I only send values if they have changed beyond a certain amount of their previous value. If you were to capture this serial data it might look like...

%100P%120P%140P@%200T%160P

... where '%' indicates the start of a data block and the 'P' or 'T' indicate that it was Pan or Tilt data. Oh, it only supports Pan and Tilt, not Roll, but this could be easily implemented if desired, I just don't have any models with Roll.

## SoftwareSerial

I experimented with this for a bit, but found that the receiver would often not get the correct data and so I tried adding a checksum to each piece of data so that I could reject them. Whilst that worked, the drop rate and SoftwareSerial processing overhead on the head tracker unit significantly reduced the performance.

Rather than remove the SoftwareSerial code, I left it in place controlled by an ‘#ifdef’ directive so that if anybody else really wants to play with it, then they can.

## More Bluetooth Info

So, if you try to configure the pairing of version 3 devices using the version 2 steps you will come across a step that does not work. The AT+INQ command gives you an ERROR(1F) response. This error code does not appear in the version 2 documentation.

The post by Presidenten describes two different command modes and how you can switch between them to issue the various AT commands. However, following his process I still found one or two that appeared to need to be run in ‘the other mode’ so as not to return an error.

To be honest, I'm not entirely convinced my HC05 got configured 100% as expected. The reason I say this is that, if I powered up another HC06 (with same password), then the HC05 would still pair with it. But if I powered up my 'preferred' one as well as any other random ones, then the HC05 would pick the 'preferred' one over the others. This led me to the conclusion that if you just set matching passwords, then the default behaviour of the HC05 will allow it to find and pair with an HC06 without any of the configuration described by Presidenten.

It is perfectly feasible that you could swap out the Bluetooth modules for the very popular ESP wifi modules instead.

The internet community can be a wonderful thing and there are plenty of other far cleverer people than me out there also struggling with the same problem, and so eventually, over time better documentation will become available.