# Pi In The Sky Telemetry Board

It’s not that onerous a task to build and code your own tracker, but not everyone has the time, or may have the software skills but not the hardware skills (or vice versa). Although some schoolchildren as young as 15 have built and coded their own trackers, for the most part it’s more practical to buy a ready-made tracker. It was for these reasons that we designed the Pi In The Sky (PITS) board.

This board combines 3 essential devices that any balloon tracker needs - GPS, radio and power supply. It also includes a temperature sensor and an ADC to monitor battery voltage and current, plus it can be extended with external I2C sensors (e.g. pressure) and an external temperature sensor.

Hardware features are:

* Efficient built-in power regulator providing run time of over 20 hours from 4 AA cells
* Highly sensitive UBlox GPS receiver approved for altitudes up to 50km
* Temperature compensated, frequency agile, Radiometrix 434MHz radio transmitter
* Temperature sensor
* Battery voltage monitoring
* Sockets for external i2c devices, analog input, external temperature sensor
* Allows use of Raspbery Pi camera
* Mounting holes and spacers for a solid connection to the Pi

Of course hardware is only part of the solution, so the board is provided with open-source tracker software which is configurable so you can control how it operates.

* RTTY Radio telemetry with GPS and sensor data using UKHAS standard
* Radio image download using SSDV standard
* Multi-threaded to maximize use of the radio bandwidth
* Variable image size according to altitude
* Stores full-definition images as well as smaller transmitted images
* Automatically chooses better images for download
* Configurable via text file in the Windows-visible partition of the SD card
* Supplied as github repository with instructions

## Assembly

Assembly is very straightforward using the supplied standoffs. Remember to connect the camera cable **first**, and using a special GPIO socket with extended pins if you are going to stack another board on top.

## Software Installation

First, download the latest Raspbian image from the RPi download page, following the instructions on that page. The following instructions are for the latest version of Raspbian (May 2015) and we strongly recommend that you download and burn that to an SD card, 8GB or larger (8GB is plenty unless you want to store a lot of video).

### Install Updates

Insert the SD card into a Pi model B, connect to your LAN, and apply power to the Pi. There’s no need at present For the Pi board to be connected to the Pi In The Sky (PITS) board, however it does need an internet connection. We strongly recommend using a model B for development (so easy internet access) and a model A for flights (so a lot less power required).

If you're using a keyboard and monitor, then raspi-config (a configuration program) will start up automatically. Otherwise, you can connect to the pi using an ssh client (e.g. putty) provided you know its IP address. Login using user “pi” and password “raspberry”, then type:

sudo raspi-config

Either way, with raspi-config running, you need to set a few things:

* Expand Filesystem
* Enable Camera → Enable
* Advanced Options --> SPI → Yes → Ok → Yes → Ok
* Advanced Options --> I2C → Yes → Ok → Yes → Ok (if you will at some time use the BMP085 or BMP180)
* Advanced Options --> Serial → No → Ok
* Advanced Options → Device Tree → No → Ok (this is only needed for Raspbian 30th Jan 2015 or later)
* Select Finish → Yes

Then close raspi-config and shut down ((this is needed so that the file system is expanded, otherwise the next steps will fail):

sudo halt

then connect the PITS board and camera, and connect the GPS antenna to the GPS socket. See this page for the connections for your board (PITS or PITS+). Connect to power again (either 5V to the micro USB power socket, or 4 AAs to the power connector on the PITS board).

When the Pi has booted, login again then run these commands:

sudo apt-get update

sudo apt-get upgrade

The second command will take some time.

### Wiring Pi

Next, install the excellent "Wiring Pi” software from Gordon Henderson. This provides a command-line app “gpio” which we can use to test the PITS board, plus some C libraries which the PITS software uses to access the PITS hardware. Install using these commands:

cd ~

git clone git://git.drogon.net/wiringPi

cd wiringPi

./build

### Install SSDV Software

Now install the equally excellent SSDV (Slow Scan Digital Video) software from Phil Heron. This provides a command-line app which converts between JPG and SSDV formats. In our case we convert from JPG to SSDV and then transmit the resulting packets over the radio link. Install using these commands:

cd ~

git clone https://github.com/fsphil/ssdv.git

cd ssdv

sudo make install

### Tracker Software

The PITS tracker software is on github, and can be installed as follows:

cd ~

git clone -b LoRa https://github.com/PiInTheSky/pits.git

cd pits

./build

The build process compiles and links the tracker program, creates a default configuration file, and sets the software up to start automatically when the Pi boots.

### Testing

With the camera and PITS boards connected, run these commands:

cd ~/pits/tracker

./startup

You should see the red WARN light come on straightaway. This will stay on until the GPS receives lock, which first time could take 2 minutes or so. If after 5 minutes there is no lock, move the GPS antenna to a window, or outside. The receiver is very sensitive and you should get a lock anyway so long as it has sight of a window.

Once GPS lock has been established, the WARN LED will go out and the green OK light will flash.

Meanwhile, on your monitor screen (or ssh window) you should see plenty of activity. You should not see any errors – if you do then read the message carefully and see where you went wrong in following the instructions above.

At this point you should set up a radio receiver to decode the transmissions.

When you are done testing for now, you can shutdown the Pi with "sudo halt".

You do not need to start the tracker software manually again - it will automatically start when the Pi has finished booting.

### Updating The Software

Periodically we update the software on github, and we do recommend that you always use the latest available.

To be informed of any updates, you need a github account, after which you can go to https://github.com/PiInTheSky/pits and "star" the repository.

Alternatively, follow https://twitter.com/pitsproject on Twitter.

To download and build the latest software, perform these commands either at the Pi keyboard or in an ssh session:

cd ~/pits

git pull origin master

cd tracker

make

## Editing The Configuration

On the Pi, type

sudo nano /boot/pisky.txt

and you will see the configuration file:

payload=PISKY  
disable\_monitor=Y  
frequency=434.300  
baud=300  
camera=1  
low\_width=320  
low\_height=240  
high=2000  
high\_width=640  
high\_height=480  
image\_packets=4  
logging=GPS,Telemetry  
#info\_messages=0  
#enable\_bmp085=N  
#external\_temperature=0

You can change these values:

payload – This is the name of the flight, as shown on the live map. Keep it to no more than 6 characters if you are going to transmit images

disable\_monitor – Set to “Y” to disable the monitor outputs, saving approx 20mA @ 3.3V. Do NOT use if you use a monitor connected to the Pi!

frequency – Used to set the radio transmission frequency, in MHz.

camera – Set to 1 to enable the camera transmissions, or 0 to disable them. With either setting the camera will still be used for hi-res stored images (assuming a camera is connected). Only the Pi cameras (standard or Noir) are supported

low\_width/height – the dimensions of images taken at low altitude. It’s best to keep the same aspect ratio (1.333), and both values must be multiples of 16.

high – altitude above which the system uses the high\_ settings instead of low\_. in other words, it allows the tracker to take larger images once airborne

image\_packets – This is the number of image packets sent for each telemetry packet. “4″ provides a good compromise between image throughput and the delay between telemetry updates

enable\_bmp085 - Omit the line, or set to "N" or "No", to disable use of a BMP085 or BMP180 pressure sensor. Set to "Y" or "Yes" to enable. This device is not included on the board. Instead the board has I2C connections (Vcc/GND/SDA/SCL) for connection to an external BMP085/180 device. See this page for details.

logging - this controls what gets logged to the SD card. Options are "GPS" and "Telemetry", and create gps.txt (positions from GPS) and telemetry.txt (radio telemetry sentences).

enable\_bmp085 - Enabled the code for the BMP085/180 pressure/temperature sensors.

external\_temperature - Sets which DS18B20 is used for external temperature measurement.

info\_messages - sets the number information messages sent over the radio when the program starts. Default (i.e. line missing or commented out) is 4 for baud rates of 300 or higher, and 2 for lower baud rates.

## Deleting Images / Log Files

Create the file /boot/clear.txt, either from the Pi (sudo vi /boot/clear.txt) or by inserting the SD card in a PC. The contents of the file do not matter.

Next time the tracker program is started, the image files and log files will be deleted, as will the clear.txt file itself. i.e. it's a one-shot use. This prevents the images from being deleted again if the Pi is accidentally rebooted.

The deleted files are:

SSDV images (/home/pi/pits/tracker/download/\*.jpg)

Full-size images (home/pi/pits/tracker/keep/\*.jpg)

GPS log file (home/pi/pits/tracker/gps.txt)

Telemetry log file (home/pi/pits/tracker/telemetry.txt)