Setting up a FrSky

Taranis X9D plus Transmitter,

an X8R (or other X-series) receiver,

servos, telemetry,

and the S.BUS interface

**(and now with added
868 MHz)**

Peter Scott

Last edit 28 March 2024 Version 1.4.34 (see last page for version changes)

Table of Contents

I wrote this manual over a long time period. I kept adding bits as I learned about them. As a result for some areas it doesn’t have the optimum structure. Your best approach to finding something out is to Ctrl-click on the contents list as the first step. Maybe one day I will do a major rewrite but that will take up precious flying time, so maybe not.

Beginner’s note

Even if you are a beginner you should have no trouble using the Taranis to set up a standard model. Just follow the steps under the heading on page 8 **Creating a new standard model**. You could just jump to that now and come back and read the General notes later. When you have got the first model working you can try the more advanced functions.

It is best to join a club. If you don't want to do that and you live in the UK, please join the British Model Flying Association as a country member for the insurance: www.bmfa.org I have tried to explain all of the special words used as I have gone along. There is a glossary if you are still baffled by one.

General notes

OpenTx and FrSky

FrSky (pronounced Free-sky) decided to use **OpenTx** in same way that phone companies use Google Android. Note that OpenTx is open source software and so under constant change and improvement. By the time you read this it the screen appearances might be different but the methods are likely to be the same or very similar. Open source is a way of developing software collectively at a distance, with volunteers contributing what they are best at. The resulting software is usually free of charge and often better than that sold by companies. However you must get used to the documentation never being completely up to date and being rather terse and techie. Improvements are frequent and it pays to keep the firmware in your transmitter, receiver and other devices up-to-date. If you find strange things happening it should be the first thing to try.

FrSky’s devices use a frequency-hopping protocol that allows many users to use the same band of individual frequencies. The transmitter/receiver pair can switch frequencies if they experience interference or loss of signal. Frequency hopping was invented by the film actress Hedy Lamarr in the 1940s, initially for torpedo guidance resistant to German jamming. The version used by FrSky is called ACCST (Advanced Continuous Channel Shifting Technology).

I like to write things down when I'm learning (which I now discover is called the ‘Feynman Method’), so I thought I'd assemble my notes into a manual in case anyone else finds it useful. It’s also useful when I forget how to do the less common things, especially when I am out and can read it on my website. For speed I have used **Tx** for transmitter and **Rx** for receiver throughout.

OpenTx is very flexible. The more you understand about it the more you can do. It is simple to set up a standard model. A wizard guides you step-by-step through the choice of model type and channels. It is no more difficult than any advanced Tx.

I don’t claim to be an expert. The methods in here are how I’ve done it for fixed wing models. There might be another way that is better. If so please let me know and I will add it and give you credit for the change in a footnote. Please don’t suggest online videos. I find them clumsy and inefficient.

OpenTx Companion

This is a software simulation of the Taranis Tx. You can download it free from http://www.open-tx.org/downloads.html and use it to try out setting up systems. The Taranis Tx screens and words are often different from those on Companion so I think it is best to do all experimenting on the Tx, ideally using a test bed as described later. Then you will know how to make changes on the field, like minor adjustments to servo throws and centring.

Taranis transmitter

The great strengths of the Taranis Tx are the speed and convenience of binding and failsafe, its excellent range, the built-in telemetry and the outstanding range of small telemetry sensors available. Plus, of course, the very modest prices. You can use the standard setups but you have available great flexibility. On the carrying case it says, 'Let you set the limits'.

When setting up the Tx there are two work areas. The first is **Radio setup** which applies to all models and is selected by a long press on **MENU**. The second is **Model setup** which is done for each model. A list of pages in each is at the end of this document.

There is only one momentary switch. It is often used as a buddy box switch so not available for others functions like instant trim. I might change one of the toggles to be a momentary one.

The Taranis is an excellent transmitter. However it is always work in progress. There are a few rough points, which I have described at the end, but these are always under review and improvement.

Manuals and help

This is a weakness of the open nature of OpenTx and FrSky. The device manuals are generally brief and sometimes unclear. Even powerful and complicated devices only get two sides of A4. Mostly it doesn’t matter, but you might need sometimes to get help from a forum or be confident enough to experiment and figure it out yourself.

Please donate to OpenTx

One of the reasons that FrSky is low cost is that the software is written by volunteers. However there are still costs, such as storage. If, like me, you make a lot of use of OpenTx please consider making a contribution from time to time. I do it once a year. Go to https://www.open-tx.org and choose the Donate option. You can pay by card or Paypal.

Functions, data, buttons, templates and navigation

General

OpenTx is big. To give step-by-step instructions for each function would be impracticable. The underlying techniques for selecting and editing data are common to all. Initially then, let's learn and practice these techniques with a practical exercise starting on page 8. For each function, such as binding or servo setup, all I then need to describe are simpler instructions, sample lines and data, an explanation of how it works and any things unique to that function.

Functions

In OpenTx a function is a line of code that does something, for example cutting the throttle when a switch is moved. There are two kinds of function: **global** and **special**. Global functions apply to all models and special functions apply to one model. You set both up in the same way, but some model-specific functions can only be special as they might be dangerous. There is room for 64 of each **-** GF1 to GF64 for all models and SF1 to SF64, with one set for each model. As you work, think whether commonly used functions like buddy box or throttle cut should be done as global functions. Another global function might be screen shot to enable you to keep data, perhaps GPS position if you lose your model or evidence of a personal best speed. A function can be set up on any function number. You don't have to use the function numbers shown in the sample lines here. You can substitute any unused one.

Data

Data stores are called variables. OpenTx lets you store a value that you use frequently in a **Global variable** (GV). You then choose it by name (GV1 etc) rather than typing in the value each time. Changing the value changes it everywhere it is used. What's more you could set up a function to allow a rotary control to vary the value of a GV. I use the possibly archaic word ‘datum’ to mean one item of data, a word that is plural. I haven't found a need for a global variable yet.

Transmitter buttons

EXIT	This takes you back a step or a page
+	This moves you one way along a line, through a page or through a choice list
-	This moves you the other way along a line, through a page or through a choice list
ENT	This opens a datum for edit or accepts an entered datum value or other choice
MENU	See below
PAGE	See below

Templates

Once you have set up a model and found the settings that suit you, make a copy to a new model called ‘**Template**’. Then you can use this setup for all future models by making a copy of Template. It will save you entering lots of code. You will probably need to make adjustments for the new model but these are likely to be minor.

X8R receiver aerials

The latest X8Rs have thinner aerials without a plastic box round them. I put heat shrink tubing on them as otherwise Velcro won’t stick to them. In the latest upgrade to RX8R the aerials are a simple wire with a bulge. This makes them a bit more difficult to anchor, but I now glue in two lengths of fuel tubing and push the aerials into those.

Inputs, mixer and outputs and other uses

These three core model setup pages on the Tx shape the progress of your control instructions from the stick to the servo.

Inputs

On this page you define how the Tx deals with the movements of sticks and rotaries. Sticks are shown as ▯ and rotaries as ▯. Inputs can be named and are shown as I in later lines of code. You can use a trim as an input. Though unlikely, for completeness these are shown with a symbol similar to ▯ with a vertical spike top and bottom.

Mixer

Here you can combine (mix) more than one input to control one output servo. For example you can mix in a small amount of flap signal with the elevator signal to push the nose down when you lower the flaps. Again you can name mixes and use them in later lines of code. There is an example of more complicated mixes at the end of this manual.

Outputs

On this page you define how the Tx controls the servos and other devices. For example you can set throw, expo, centring and direction. There are eight output channels to the sockets on the Rx, though the Rx is capable of sixteen channels. If you need more than eight you use a second Rx set up for channels nine to sixteen. Alternatively you can use the single S.BUS socket to connect up to sixteen channels on one Rx.

Other uses

I am sure I have seen a Taranis used on Robot Wars. I have worked with someone who is using one to control a large model tank. No doubt it will have been used for yachts and other floating models. However there are no built-in templates or dialogues for such models. You will have to set it all up from scratch. It will pay you to build a test bed of some sort on which you can wire up and experiment with the bits before you fit them in your model. Later on in this document I describe one that I built for fixed wing aircraft.

Live changes

When adjusting control surface weight (throw) any changes you make work immediately. If you have a servo that is buzzing due to too much weight, hold the stick or rotary on full throw and reduce the weight until the buzzing stops.

Navigation

When you switch on the Tx you see the **Home screen** displaying one model.
From here there are three routes through the Tx using the **MENU** and **PAGE** buttons.
Route 1: **Current model settings**: Press **PAGE** to move through pages
Route 2: **Radio setup**: Long press **MENU** then press **PAGE** to move through pages
Route 3: **Model setup**: **MENU** for list of models. Select model. Press **PAGE** to move through pages.
There are lists of pages in these three routes at the end of the manual.

Creating a new standard model

If you want a standard setup for your model it is a simple matter to set it up on your Tx. Standard means four channels with both ailerons on one channel and no flaps or airbrakes.

It is a good idea to stick a unique number on the receiver that you are going to use.

Switch the Tx on
Press **MENU**
Using **+** and **–** scroll to a blank line.
Hold down **ENT**
You will see **Create model** highlighted.
Press **ENT**

You get a row of pictures showing an airplane, a flying wing and a drone.
The airplane is highlighted so press **ENT.**
You get the message **Has your model got an engine?**
Yes will be highlighted.
There is a line to the picture of the motor labelled **CH1.**
As you want throttle on channel 1 just press **PAGE.**
Has your model got ailerons?
Yes, 1 channel
CH2 has been assigned to ailerons
(If you want the ailerons each on a different channel press **ENT** and select **Yes, 2 channels**. Press **–** to move to the channel numbers and change **CH6** to **CH5.**)
Press **PAGE**
Has your model got flaps
No will be highlighted
Press **PAGE**
Has your model got air brakes?
No will be highlighted
Press **PAGE**
What is the tail configuration on your model?
Ele(1ch) = Rud(1ch)
CH3 is assigned to elevator
Ch4 is assigned to rudder
That is a common default so press **PAGE**

You then get this screen:

Ready to go?	
Throttle	**:Ch1**
Ail	**:CH2**
Rudder:	**:Ch4**
Elev	**:Ch3**

Hold down **ENT** to confirm that is what you want

You now see that a new model has been created, for example:
***4 MODEL3 187** (numbers might be different)
The asterisk means that it is the selected model that you can now edit or fly.
The name can now be changed.
The **187** is the size of the data file used to store the model details.
As you edit the model further this will change.

There are a few more things to do.
Check that your new model is still asterisked.
Press **PAGE**
You are now on the **MODEL SETUP** page
The first two items are **Model name** and **Model image**
Leave these for now.
Using **+** and **–** scroll down to **Receiver No**
A number will be highlighted
Press **ENT** and it will flash
You use **+** and **–** to change it to the number on your receiver.
Press **ENT** to store it
(If you are warned that you have used this before press **EXIT**).

Now you must do the following things. Instructions are in a few pages:

- Bind the receiver to the transmitter
- Set the failsafe, probably just setting the throttle to zero
- Set a switch to cut the motor off if you are flying a model with an electric motor
- Edit the name of the model to something useful
- Choose a picture of your model for main screen

Only the first two are required by the BMFA for safety and insurance reasons.
The third is crucial for your safety.
The last two can be left until later.

How to enter or edit individual data

It is best to create a new model as you are certain to make mistakes. It is even better if you use a test bed as described later.

Let us learn how to set a timer
At the **Home screen** press **MENU** to get model list
Move to a model or create a new one (above)
Press **ENT** to select
Press **PAGE** to enter **MODEL SETUP**
Move down to **Timer 1** using **-** button
Press **ENT** to select
You are now in a datum box which is solid so is not ready to edit

Press **ENT**
The box is now flashing to show it is ready to edit. It displays **OFF**
Press **+** or **–** to change it to **Ths.** Counts down when the throttle is more than zero
Press **ENT** to store value
Press **–** to move to the minutes display which shows solid **00**
Press **ENT** to open it for edit. It will flash
Press **+** to set the number of minutes
Press **ENT** to store value
Press **EXIT** repeatedly to get back to the home screen
This method applies to all data entry with very few variations

So the pattern is:
Move to the chosen datum using **PAGE**, **+** and **–** as required. Datum is solid
Press **ENT** to edit. Datum flashes
Use **+** and **–** to find correct value
Press **ENT** to store value
Press **EXIT** to leave entry
Wherever possible this will be written something like '**Move to Page/datum XXX (and open for edit)**'.

Copy a model

From the **HOME** screen select a model
Hold down **ENT**
Select **Copy Model**
Move to a blank line
Hold down **ENT**
A duplicate will appear This is useful when editing. You can delete your mistakes and go back to the copy.

Backup a model

Do as **Copy model** but select **Backup Model**
The model data will be copied to the SD card.

Restore a model

This means copy a model from the SD card into the list of models
In a line for a blank model, hold **ENT**
Select **Restore model**
Scroll to the correct model
Press **ENT**

Delete a model

You can't delete a model that is currently selected. Select a different model. Return to the one you want to delete and give a long press on **ENT**.
Select **Delete Model**.

The microSD memory card

The microSD card slot is in the battery bay. The card has no special format, just folders and files. It is sensible, after each major change, such as a new model, to backup (archive) the model to the card and plug the card into a card reader in your computer. Then back up the files to a dated folder using **Copy** and **Paste**. You can then copy them onto a second card as a backup.

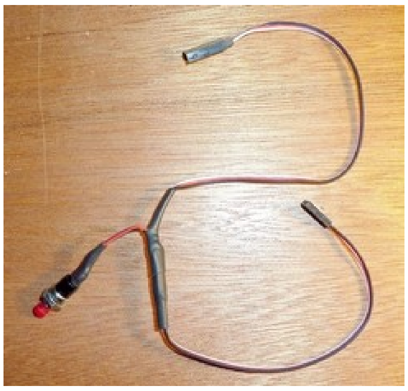
To view the files, enter **RADIO SETUP** and move to the **SD CARD** page. Of the eight folders these are the four that you are most likely to use:

BMP This is where you should place the 64x32, 4-bit grayscale .bmp files that you want to use as model logos. Filenames must be 10 chars long or fewer, not including extension .bmp. You can convert suitable digital pictures using Paint and Photoshop
LOGS This has the telemetry logs, if enabled. Data is stored in an automatically created file with the same name as the model and the date. One log file is created per day for each model.
MODELS Model files saved by the **Backup model** command of the model selection screen go here.
SOUNDS Put new voice packs here. ZIP files with the standard voice packs can be downloaded onto your computer from FrSky's website. ZIP files have to be expanded (extracted) and copied to the SD card.

Binding

You need a softly pointed object such as a pen to push in the recessed Rx button **F/S**. Binding is instant and the easiest of all radio control makes. Note that the Tx should not be closer to the Rx than about one metre when binding due to signal swamping. You might get a message that this receiver number has already been used. Press **EXIT** if you are sure you are doing the right thing.

Select the correct model
Move to **MODEL SETUP** page
Select the correct Rx by number (I stick a number on each Rx)
Move to the **[Bind]** box
Press **ENT**
You must then choose a channel range (usually **Ch1-8 Telem ON**)
Press **ENT**
The screen box flashes and the Tx beeps.
Hold in the **F/S** button on the Rx and apply power to one of the servo ports.
The green LED is on and the red LED blinks to show bind is complete
Release **F/S**
Switch off Rx
EXIT from Bind mode. Switch on Rx
The green LED should be on and the red LED off



This is tricky if you are using an ESC for electric flight as *you don't have three hands. I made up a lead, using a long servo extension lead by cutting the red wire, soldering in a momentary push switch and insulating with heat shrink. I then coiled up the unbroken black and the cut white wires inside some heatshrink. I can now connect the ESC, then push in the F/S button and press the switch.*

IMPORTANT NOTE: If your transmitter and receiver are capable of failsafe the BMFA insists that you use it. It also expects you to test for failsafe and range on each model before you fly in each flying session.

Range check

The X8R receiver has **Received Signal Strength Indication (RSSI)** telemetry built in. You do not need to set it up. Range check simply involves setting the Tx to **Range** check. This reduces the signal to 1/30th (about -14.5dB), and you then watch the screen as you walk away. The received signal starts at nearly 100 when close to the model. As you move away it drops. You will find you will be at least 30 m away before it falls to the warning level of 45 and you get a voice warning. You can, of course, also move the control surfaces in the usual way. Users of other makes of Tx are baffled when you don't.

Switch on the Tx
Connect the ESC to the battery
Select the correct model
Move to **MODEL SETUP** page
Move to [**Range]**
Press **ENT**
A new screen pops up with the current RSSI value on it and the Tx beeps
Walk away till it drops to 45.
EXIT

Incidentally you can use RSSI to find a lost model provided the Rx is still powered. Display RSSI on a telemetry screen or switch to range check. Walk to where you think it is. Once you get an RSSI signal you know it isn't far away. Then move and watch the signal increase or decrease. Turning round so your body blocks the signal will tell you direction as well. We have found models using this technique.

Alternatively as you realise a model is going down or out of sight you could make a screen shot of the telemetry screen showing GPS data. Then feed the map reference into Google maps in your phone or your satnav. I haven't tried this but I think the precision of the data should be good enough.

Setting up failsafe

This is very flexible.
Select the current model
Move to to the **MODEL SETUP** screen
Move to **Failsafe mode**
Open [**Not set]** for edit
Change it to **[Custom]**
Scroll to **[Set]**
Press **ENT**
You get a screen showing a horizontal bar for each input like sticks and rotaries. By scrolling, set each input to a value between -100 and +100, which is fully one way or the other. Most should be 0.0 except throttle which will be -100.0, unless, for example, you want to set flaps that only require half throw.

Now transfer the values from the Tx to the Rx:
Switch on the Rx
Push in the Rx **F/S** button.

Note that in V2.2.2 failsafe does not trigger automatically. You are warned that the signal has been lost and have to press **ENT** to go into failsafe mode. It is therefore important always to have sound on at an audible level.

Just one warning. I set it wrongly at first. When I opened the set failsafe screen I thought the dotted bar showing the throttle stick position was the final setting so I didn't set it. The motor roared when I switched off the Tx to check failsafe. The set value bar is solid and appears as you change the number.

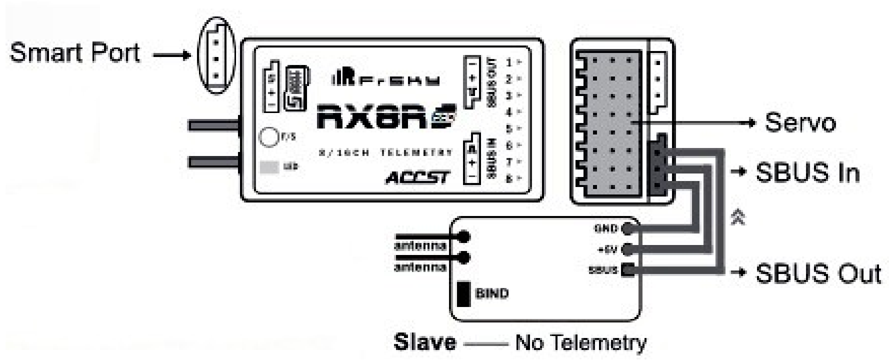
Slave receivers (redundancy)

A slave receiver allows an even more reliable conection between the Tx and the model. If the master receiver loses the signal and the slave still has it you will not go into fail-safe and will still have complete control. To be honest range is rarely a problem with FrSky eqipment, provided you keep the aerials away from each other and things like servos. Having one vertical and the other horizontal is good idea.

The master receiver manual must state that it allows (supports) redundancy. A good example is the RX8R Pro. You need a second (slave) receiver with an S.BUS output, for example XM+, XM, R-XSR, L9R, etc. FrSky recommends XM+. Do not use a second RX8R as the slave as telemetry will not work.

If there is telemetry on the slave you must disable it. This is an option when you bind it. Set the slave to channels 9 – 16 output with no telemetry.

The diagram shows the connection. A female to female servo lead is used to to connect the S.BUS sockets.



Throttle cut

This is a vital function for an electric model. Unlike an internal combustion engine, a motor will start with no warning if the throttle stick is moved accidentally. The following shows you how to create **Special functions** for throttle cut. It could also be done as **Global functions** which would then apply to all models but the switch cannot then be used for anything else. I use switch SC which has three positions. Taranis only has two two-position switches. I use one for rate and the other for buddy box. Here we will set up two of the three positions to trigger throttle cut and tell us that it has happened.

Select the correct model
Move to the **Special functions** page
Move to the next free **SF** and open for edit (It will be SF1 if you haven't used any yet)
Press **ENT** so the **---** is flashing for edit
Move switch **SC** to its middle position
You will see **SC-** flashing
Press **ENT**
Leave **OverrideCH1** as this what we want (Ch1 is throttle)
Open the number datum box for edit
Change the number to -100
Change the empty box to a ☑
EXIT back to main model screen
Push switch **SC** away from you
Move to **Channel monitor**
Move the throttle to maximum
Move **SC** to the middle position and the throttle drops to minimum

Now create a special function to do the same for switch position **SC↓**
It is useful to have a voice message to tell you the throttle is cut.

Move to **SF3** (or the next unused one)
Again select the mid position **SC-**
Move to **OverrideCH1** and open for edit
Move to **Play Track**
Move to **Beep1** and open for edit
Move to **Play track** and open for edit
Move to **---** and open for edit
Scroll down to **engoff** (“Engine off” built-in message)
You only want it said once shown by **1X** so leave that alone
Do the same in the next SF for the other switch position

It now should look like this:

SF1	**SC-**	**OverrideCH1**	**-100**	**☑**
SF2	**SC↓**	**OverrideCH1**	**-100**	**☑**
SF3	**SC-**	**Play Track**	**engoff**	**1X**
SF4	**SC↓**	**Play Track**	**engoff**	**1X**

If you move switch **SC** you will hear the message for all but one position

Move to **Channel monitor**
Play with the throttle stick and switch SC to make sure it all works correctly
Unfortunately there is no default message for engine on. You can transfer one called ‘engon’ from the sound pack. You would then have to add a further Special function for **SC↑** that plays a track “engine on” or perhaps 'Throttle high'.

Setting up the servos

For this you use the **OUTPUTS** screen so move to it. You reverse servos on this page and adjust servo centring. You might also use this to set up different up and down throws on ailerons.

There is a line for each output, normally an ESC, a servo or retract motor.

OUTPUTS 988us 7/12

Channel	Name	Subtrim	Min	Max	Direction	Curve	PPM centre	Subtrim mode
CH2		**0.0**	**-100**	**◄100**	**►**	**---**	**1500**	**▲**
CH3	**etc**							

Most of these need not concern us at this stage.

The key data to edit are:

Subtrim	Servo centre point. Leave at 0.0
Subtrim mode	Leave on ▲ (see below)
Min	Set a number more than -100 to reduce servo throw (NB -70 more than -100)
Max	Set a number less than 100 to reduce servo throw (can be different from Min)
Direction	To reverse a servo click ENT

As you edit values in the usual way, the system changes immediately, so you can experiment with the sticks. Note that if you have each aileron servo on its own channel you can set differential with more up movement than down.

A little more needs to be said about **subtrim modes**.

Default ▲ This moves the centre of the servo movement. Therefore the servo will move further in one direction than the other.

Mode = After the centre adjustment the throw each side is set to be the same. This can produce too much throw on one side and cause clipping and buzzing. This throw would have to be reduced. This mode is only needed where differential or mixing is used for example on separate aileron servos.

Setting up dual rates

Decide what switch and switch position you want for each rate (weight). You will probably have only low and high rate, but for highly aerobatic models you might have low, high and very high (3D). The model's plan or manual will tell you what movement or angle throw to use for each rate. Set the weight for the highest rate to 100 first and adjust the mixer or outputs to give the correct high throw. If you need to reduce the weight more than say 20 then you should move the control surface rods to different positions on the servo arm and control horns first. Then experiment with the weight(s) for the other rate(s).

Select the correct model
Move to the **INPUTS** page
Move down to the selected control surface, eg **Ail**
Press **+**
You now get an extra line for **Ail**
Move down to it
Long press **ENT** and select **Edit**
Edit the curve parameters (Weight 0 – 100, Expo)
NB: if **Expo** is zero the servo moves linearly with the stick. If it is positive the servo moves less to start with then more sharply. This is useful for giving gentle control movements.

Move down to **Switch**

Select one of the switches by name and position eg **SF↑** or **SF↓**
Then Move back to the original **Ail**
Hold down **ENT**
Select **Edit**
Move down to **Switch**

Select the same switch by name and choose the position not used above eg **SF↑** if **SF↓** used.
If you now operate the switch you will see the relevant switch setting in bold so you can check the settings.
Now do the same thing for the other surfaces Ele, Rud etc.
Click **EXIT** when complete
You could now set up a voice message when switch SF is moved that says 'High rate' or 'Low rate' using special or global functions as described for Throttle cut.

Here is a sample screen for a complete setup:

INPUTS

I Thr	**100▯**	**Thr**		**---**
I Ele	**65 ▯**	**Ele**	**E30**	**SF↑**
	100▯	**Ele**		**SF↓**
I Ail	**65 ▯**	**Ail**	**E30**	**SF↑**
	100▯	**Ail**		**SF↓**
I Rud	**35 ▯**	**Rud**		**SF↑**
	70 ▯	**Rud**		**SF↓**

Aileron differential

An aileron moving up has less effect than one going down, so you might want to have twice as much up aileron movement as down. This can also avoid adverse yaw. To do this you need each aileron servo on a separate channel. It is likely that one servo will need to be reversed. However you won't want to do this on symmetric airfoil aerobatic models.

A typical setup for aileron servo 2 on channel 5 might be:

Channel	Name	Subtrim	Min	Max	Direction	Curve	PPM centre	Subtrim mode
CH5		**0.0**	**-35**	**◄70**	**►**	**---**	**1500**	**▲**

Using the mixer

If you are just starting out you won't need mixes. Before long you might want to create flaperons, crow brakes, V-tails, down elevator trim on flaps down and so on. There are no standard mixes to choose from. You have a completely free hand.

Remember that each mix applies to one control surface, eg rudder or aileron 1. You add one or more other inputs to the surface. For example you might want to add a small part of the flap input into the elevator so you get down trim to push the nose down as the flaps are lowered. Until you lower the flaps the mix doesn't change the elevator operation.

Operating the undercarriage gear

You could of course write a **Special function** to do this.
However a simple up/down can be done with a line in the **Mixer**
If the gear is plugged into channel 8 and you want to switch it with **SB** move to **MIXER/CH8**
Hold down **ENT** and you will be in the **EDIT MIX** screen
Go into **Mix Name** for edit
Using the **+** and **–** buttons select the letters for the name, perhaps **Gear.**
After selecting a letter press **ENT** to move from capitals to lower case and vice versa.
Press **ENT** to move to the next letter space and so on.
You can clear the letters by pressing **EXIT** and starting again.
Entering letters is a bit awkward but you soon get the hang of it.
Press **EXIT** when you have entered **Gear** (or a bit like it if you give up in despair)
Move to **Source** for edit
Select **SB**
EXIT back to **MIXER**

The line should look like this:
CH8   100   SB   Gear   (or gURr!)
To remove a line move to the line, press **ENT** and select **Delete.**

Elevator trim when lowering flaps

Here we want to use a rotary to operate the flaps and to add a small amount of the movement to the elevator as a down trim.
First we need to create a new input for flap.
Move to **INPUTS** and open the next unused input for edit.
Give it the name **Flaps**
Leave **Weight** as 100
Set **Source** to S1
Move to **MIXER** and select **CH3** (elevator)
Press **+** to add a new line for the flap input
Select it for **Edit**
Set **Source** to **Flaps** and **Weight** to -5
EXIT and try it out
You might need to adjust the **Flaps** weight by experiment to give the correct down trim.

Creating a flaperon

Here we want to mix the flap and aileron inputs to move the ailerons. This will give a stronger flap effect. To do this you will need each aileron to have its own channel.
Here we see how to do one aileron. The other will need the Fpon offset to be opposite.
Add a new line to **INPUTS** named **Fpon** with **Source** S2
Move to **MIXER** and **Ch4 Ail**
Add a new line with +
Move to it, Select **Edit** and add **Fpon**
Set the **Offset** to 100 (or -100)
EXIT and try it out.
The weightings and offsets will depend on the setup of your model.
Put an icepack on your head and start playing!

One more thing to note

Just one other point about mixing. Mixing might increase servo throw. If the servo weight is already set to 100, this might overdrive it. Therefore you might need to ensure that normal weight is no more than 100 minus the largest possible addition. You need to do some sums. Another icepack.

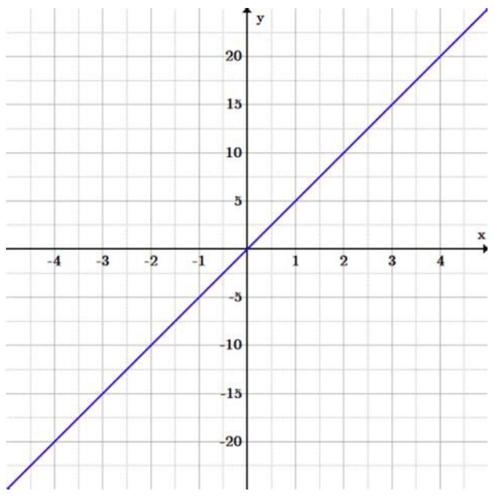
Curves including throttle curve

We are all familiar with 'expo'. This allows us to vary the effect of a transmitter control, usually a stick. We can give less control surface movement to start and more when at the extreme of stick movement, or vice versa. The first makes for more subtle control when, for example, landing and allows us to chuck our models about when we want to. However with expo we only have a single number. This is usually adequate but you have to take what you are given.

Most transmitters allow the use of more configurable curves. FrSky certainly does. In effect you can plot a graph by setting up to seventeen data points. Why would you need to? A conversation at the field made a case for using it for throttle. The proponent, Brian, was an IC enthusiast and said that his different engines had different responses to the stick. He loves aerobatics so needs precise control of response. I said that as a leccy I probably didn't need that. He said hadn't I ever noticed a lack of response up to about half throttle? I admitted that I had so decided to do some experiments, results to follow..

The notion of linear

This is a linear graph. It is a straight line, in this case going through the origin at 0,0. The maths people still call it a curve. As one quantity goes up so does the other exactly in proportion. For model control surfaces moving the stick twice as much moves the aileron twice as much. Minus values are left and down.



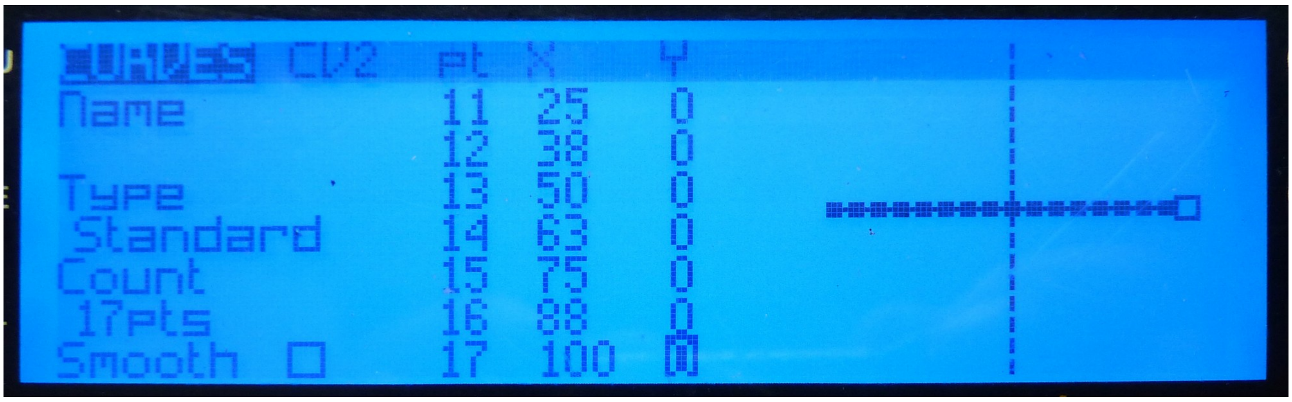
Setting a curve

This applies to any control but here we are using it for throttle. I explained above why we might need to.

To start with create a new model to play with before trying it on a real one.

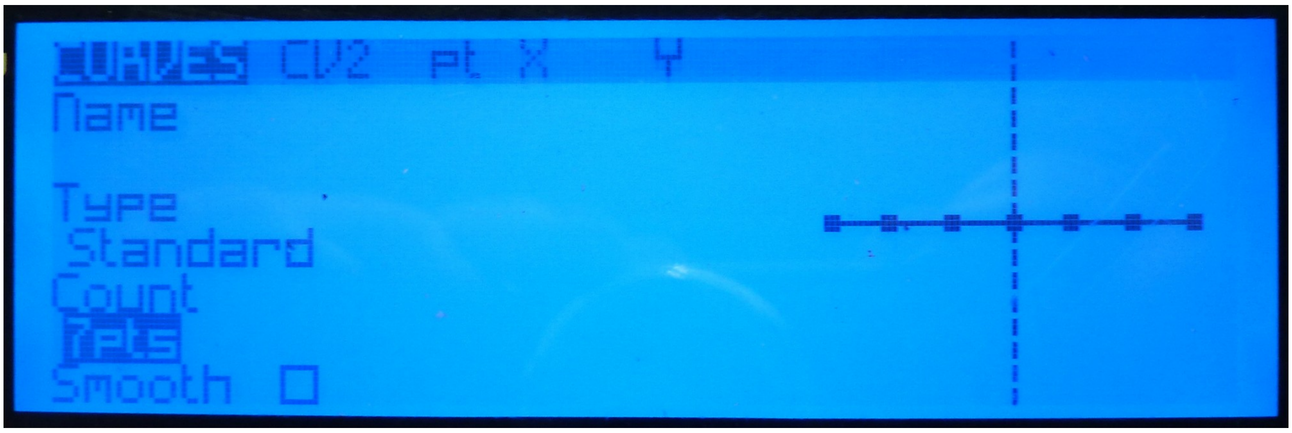
Click MENU click PAGEand select the model you created
Click PAGE six more times to get to CURVES
I am going to use CV2 (Curve 2). For a new model you will choose CV1
Press ENT

You now see a screen similar to this:



```
CURVES CV2    pt  X    Y
Name          11  25   0
              12  38   0
Type          13  50   0
 Standard     14  63   0
Count         15  75   0
 17pts        16  88   0
Smooth □      17  100  0
```

You see that you could use up to seventeen data points.
The default is five as you see on your screen but we will use seven.
Press - twice to get to Count
Press ENT
Set the number to seven and press ENT
If you choose an odd number you have a data point where the axes cross
This is how it looks now:

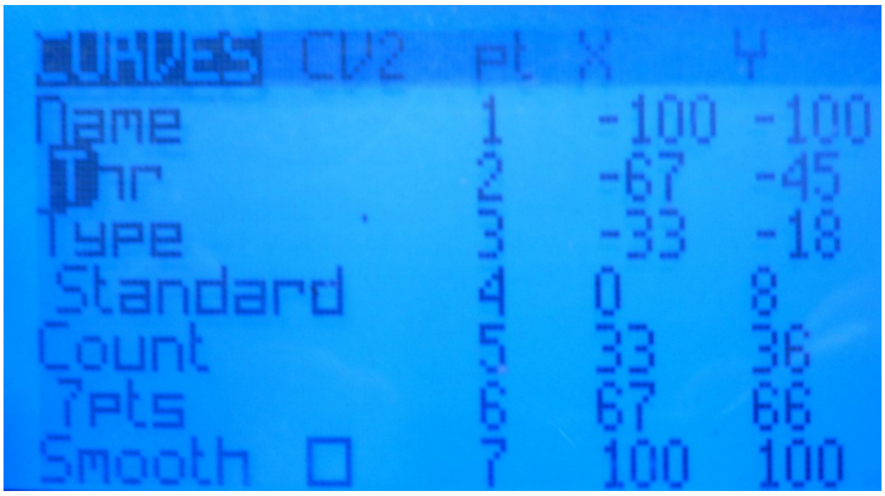


```
CURVES CV2    pt  X    Y
Name

Type
 Standard
Count
 7pts
Smooth □
```

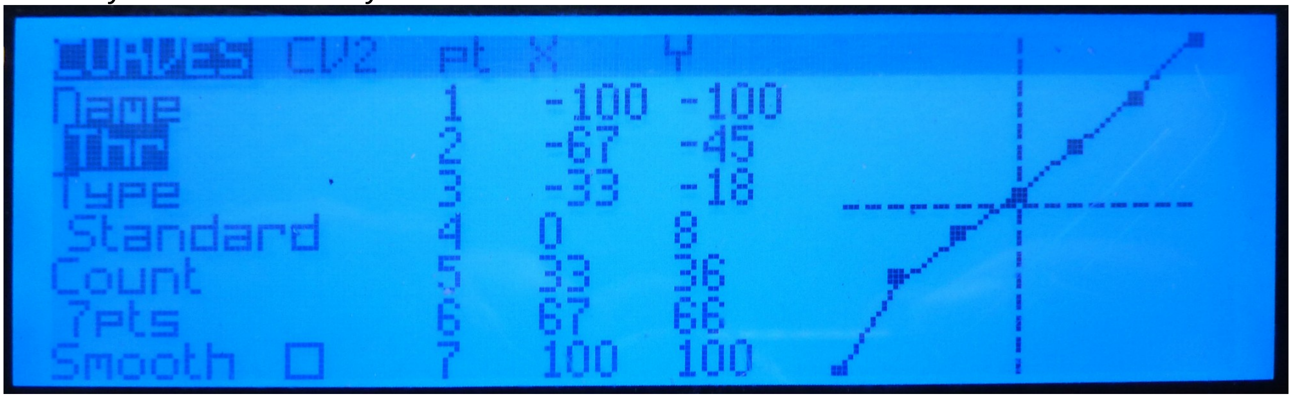
Now you can set the Y (vertical) values for each X data point.
Press ENT.
You now see a screen like the one below.
The Y data will all be 0 at the moment.
Press + or – to get to the first data point.
Press ENT and hold down – until you get to -100 as shown below.
You will see the curve change.

Set all of the data to the values shown here, or others if you prefer.



```
CURVES CV2   pt  X     Y
Name         1   -100  -100
 Thr         2   -67   -45
Type         3   -33   -18
 Standard    4   0     8
Count        5   33    36
 7pts        6   67    66
Smooth  ☐    7   100   100
```

When you have finished you will see a screen like this:



```
CURVES CV2   pt  X     Y
Name         1   -100  -100
 Thr         2   -67   -45
Type         3   -33   -18
 Standard    4   0     8
Count        5   33    36
 7pts        6   67    66
Smooth  ☐    7   100   100
```

Here the change from linear is not large. You would however see extra power coming at lower throttle stick settings.

Smooth
You can if you wish move to Smooth and set it on. Unless you have set extreme values you won't see the line change much.

Curve name
It is probably a good idea to call the curve something. I chose **Thr**. So move to the Name space and add the name in the usual way. Hold down the letter to move between upper and lower case.

Practical use
Take a model that you fly regularly. Make a Tx backup of the model settings.
Using a new or newish fully-charged battery.
Tether the model. Better still tether it with a device such as luggage weighing meter or spring balance, so you can see the thrust. Even better test the ESC, motor and propellor in the test rig that is shown elsewhere on this site.
Move the stick slowly from zero to maximum. Listen to the sound or watch the thrust readings.
If they vary significantly from linear, note the results on paper and play with curve data as described above until the response is more linear.

Recording your own voice messages and sounds

Voice messages are assembled from sound fragments stored on the SD card. There are two groups. Both are in the SOUNDS folder. In SOUNDS/en/SYSTEM there are all the sound fragments such as 'metre' and 'thirty' used, and assembled, by functions such as V**ario** and **PlayValue**. The remainder are in SOUNDS/en and are used by **PlayTrack**. The default is a female American voice. Her name is Amber. You can record your own. There is a list of the full range of messages at the end of this document. Only a few are installed by default in the transmitter. You must copy any extra ones you need to the SD card before they will appear in the scrolling list.

I am recording a new set of PlayTrack messages and storing them on the SOUNDS folder on the SD card. You must make sure you name them correctly, and that they are tightly edited so there is no silent gap at the start and finish. You can also add your own new messages. An example might be for specialist models, such as 'flame out in two' for a multi-turbine model. There is OpenTx web address for a lesson in sound management in the links list at the end. You can get complete sets of sound files. I have also installed the sound of a Browning machine gun for when I am doing fake dog fights.

First download and install the free Audacity software from Sourceforge, on: https://sourceforge.net/projects/audacity/. It will run on Linux, Windows or Mac.
You can use other recording software but here I assume Audacity.

Open Audacity
You will have a grey blank work area
Set the following:

Bottom left	Project rate **32000**
Lowest toolbar at top	**1 (Mono) Recording Channel**

Recording the sounds

Plug in your microphone or headset mike. The mike plug and socket will probably be pink. Familiarise yourself with the coloured buttons at the top:

They are the usual symbols:

●	Start recording
■	Stop recording
‖	Pause recording
▶	Playback

Do a trial run to get the level correct using the microphone slider bar. The wave should fill the box vertically on peak sound with perhaps a slight overrun.

Click ● to record the first message, for example “**Instructor has control**”
Click ■ to stop
You probably have a quiet gap at the start and finish. Remove these to speed up play.
Use the scissors **Cut** tool with click and drag to trim these away.
To check you have got it right, click ▶ and listen.
Click **Edit/ Delete** to remove the trial file (unless you got it right first time, which is unlikely)
Now do it again until you get it right.

Saving the sounds

Create a folder on your computer for the sound files
Now to save the sounds in a suitable form for the Taranis
Click **File/Export Audio …**
Move to the folder to save them in using the **Save in:** box
Make sure that the **Save as type:** box says **WAV (Microsoft) signed 16bit PCM**
Give the sound a name, for example **meinst** (the prefix **'me'** is so you can find your files in a list)
You can have up to eight characters in the name plus **.wav** of course

Now do it for all the other messages such as **gear, flaps, airbrake, low battery, throttle not zero** and **check switches** (last two on startup)

Loading them onto the SD card

There are two ways but the easiest is probably to remove the micro SD card from the Tx, put it into a carrier and insert it into your computer. It is in the Tx battery bay. The second way is to connect the Tx to the computer using a USB lead. You switch the Tx on, whilst holding the bottom trim buttons in towards the centre. Then plug in the USB lead and use copy and paste as usual.

Sound files for messages

Open the SD card
Move into the **SOUNDS\en** folder
Copy and paste the new sound files from the computer.

Changing the Welcome message for Tx startup

This is exactly like the other messages except you must call the file **tada.wav** and it must be copied into the **SOUNDS\en\System** folder. Rename the original to tadabackup.wav just in case. Make sure you select and 'eject' the SD card before you remove it from your computer.

Sound volume control

Background noise on flying sites can vary. It is useful to be able to vary the volume of messages whether from the speaker or earpieces. This is clearly a global function, so long press **MENU** and move into **RADIO SETUP**. To set the second column to ON, hold down **ENT** and select **Other**. I chose right slider **RS** as the volume control.

GF1	**ON**	**Volume**	**RS**	☑

Note: This volume control will not change the volume of the greeting message when you switch on. That must be done on the first page of **RADIO SETUP** under **Sound/Volume**.

Setting up telemetry

Here I will cover the use of the Lipo battery voltage, GPS, vario, RPM/temperature, current and airspeed indicator sensors. A liquid fuel level sensor is also available. No central controller is needed, though there is a small hub and sensors available presumably for drones. The sensors are daisy-chained using the pairs of 3 pin ports. One sensor is connected to the Rx SmartPort, then a lead from the second three-pin port goes to the second sensor. The manuals for the sensors show special SmartPort data leads but these are not needed. The supplied leads are a version of servo leads with a female connector on each end. The supplied ones are rather long so you could substitute your own shorter ones. I use 5 cm ones mostly

Apart from RPM, the sensors need no setup. Once connected each produces data. You know they are connected because the LED goes from fast flash to slower (about half a second). The data must then be added to telemetry screen(s), or combined by calculation, or set to operate a logical switch and give output, possibly sound messages.

Telemetry is set up for each model separately so it is done in **Model setup,** though beeps, tones and volume are set up in **Radio setup.** There is a full list of sensors and more technical details of the system towards the end of this manual.

Discovery

When you switch on the Rx you might find the new sensor is already in the **Sensors** list on the Telemetry page. If not move to **Discover new sensors** and click it. If they still don't appear there is a problem with the sensor or the lead.

Updates using Airlink

Sensor firmware can now be upgraded direct from your computer using a USB device called an Airlink. This is described in more detail in the telemetry section at the end.

Lipo voltage sensor

Use the JST-XH balance connector on the battery to connect to the voltage sensor. Make sure you get the black ground on the connector connected to the ground pin of the sensor. I recommend using a JST-XH extension lead. You can then put the sensor anywhere and won't keep plugging into it. You might find the pins drifting in the extension lead connectors and giving a lower voltage. I put a tiny drop of thin cyano glue onto each wire entry. Let it cure well! The Lipo sensor is quite fragile and I broke one pulling it off a velcro pad. If not already wrapped, I now wrap these sensors in the light, wide, transparent heat shrink sold for battery packs.

Switch on Tx and Rx
Select the correct model
Switch to **Engine off** if you have set that up
Move to the **Telemetry** page
Move to the list of sensors
Each has an ID
The Rx, which is also a sensor for RSSI and RxBt (Rx battery voltage), has ID 25

The list will look something like this, though shorter:

Sensors	**Value**	**ID**	
1: Alt	**1.2m**	**1**	Altitude from vario based on air pressure
2: SWR	**1**	**25**	Standing wave ratio - aerial efficiency
3: RSSI	**84dB**	**25**	Received signal strength indication
4: RxBt	**4.9V**	**25**	Receiver battery voltage
5: Vsp	**0.0m/s**	**1**	Vertical speed from vario
6: Cels	**12.32V**	**2**	Lipo battery total voltage
7: RPM	**0rpm**	**5**	Motor speed
8: Tmp1	**22°C**	**5**	Temperature of first sensor
9: Tmp2	**22°C**	**5**	Temperature of second sensor
10: GPS	**1°01'E 52°18'N**	**4**	GPS location data (not mine)
11: Galt	**30.2m**	**4**	GPS altitude (I live 29m above sea level)
12: Gspd	**0.0kts**	**4**	GPS ground speed
13: Date	**24-11-16 13:52:4**	**4**	GPS date and time

The above list was with several sensors connected on the test bed (below). There is a list sat the end showing the standard ID for each sensor:

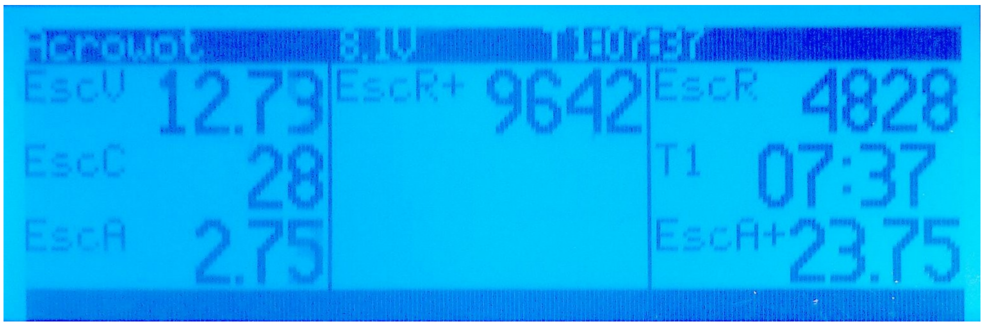
Changing telemetry device IDs

Note that these numbers can be changed using a servo channel changer as described in the section on S.BUS. The only time you might want to do this is if you connect two or more similar sensors to one receiver, e.g. two rpm or lipo voltage sensors. It looks as though the telemetry uses a connection similar to, or the same as, S.BUS.

Telemetry screen design

The FrSky Taranis allows four different screens each of which is blank until you fill it with your required data. Screen design is very easy. You move to the screen design page and first decide whether you want the screen to contain numbers or bar displays. With numbers (Pictures 2 and 3) you step to each of the nine positions and select the datum you want for that position from a list. On the bars screen you can only select three data (Picture 4), in this case just one. This is described in more detail later.

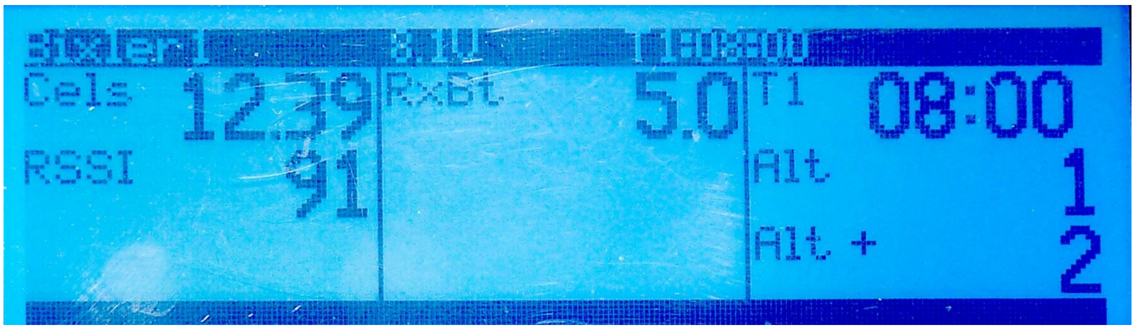
Picture 2: Numbers



EscV 12.73 EscR+ 9642 EscR 4828
EscC 28 T1 07:37
EscA 2.75 EscA+ 23.75

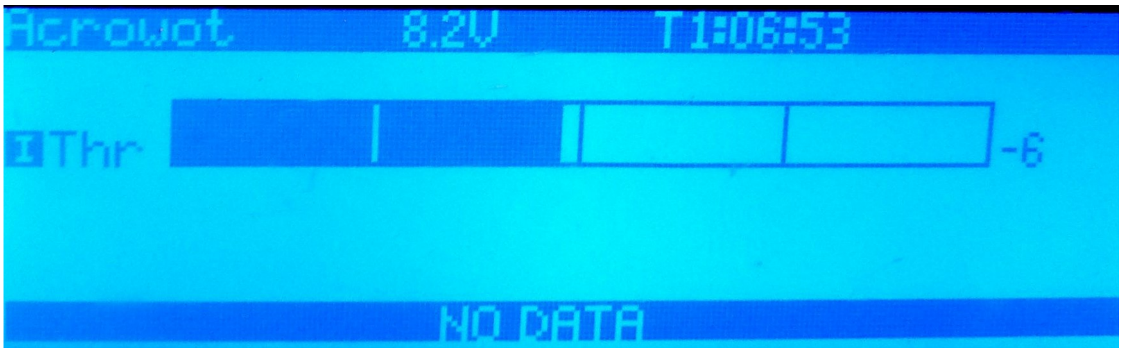
Here you can see that I have throttled up to 9642 rpm where the current was 23.75 A. Then back to low with a current of 2.75 A and rpm 4828. In playing about to take this picture I have used 28 mAh of the 2200 so the lipo still has 12.73 V. Remember that the + means the maximum for that flight. The weird names for the data are because here I am using a FrSky Neuron ESC about which more later.

Picture 3: Numbers



This is the screen for a simple glider not using a Neuron. The lipo voltage in this case comes from a separate sensor. The altitude data comes from a vario zeroed on my living room floor. RSSI means received signal strength indication, which is the signal seen by the receiver. This makes range checking very easy. 100 means full signal.

Picture 4: Bars
For completeness here is the alternative bars screen.



Telemetry voice messages

Glancing down at a screen when flying can be risky - less so for gliders than aerobatic models. Mostly I set up the Tx to speak the key data for me. Once the battery has reached about 30% charge I have the mAh used read out. For flying on a field with a height restriction I have altitude above ground level spoken every ten seconds, though I could get the message read out only when I reach 122 m. If I land and the voice says something like ‘minus two metres’ I know that atmospheric pressure is rising and the weather next day might again be good for flying.

Using the design pages

Move to the page for Telemetry screen design
Scroll down until you reach lines for **Screens 1** to **4**
Open **Screen 1** for edit
Select **None** and change it to **Nums**
You now see three columns of **---**
Each will be a numeric data box on **Screen 1**

Open the top left **–-** for edit
Hold down **+** to move through the list of data.
Note that the **+** and **–** versions are maximum and minimum values e.g **Alt+**
The new data are likely to be in a list at the end.
Move back to **Cels** and save.
Move back to the **Home screen**
Long press **PAGE**
Hopefully you now see the telemetry screen with the Lipo voltage displayed
Move back to screen design and add other data such as **RSSI** and **RxBt**
WARNING! Take care when pressing **–** when designing screens. If you switch back to **None** you will lose all of your screen setup. There is no warning.

Standard screen design

Up to now I have used different screens for different models. That can be confusing, so I decided to adopt a standard screen layout. This was my first draft:

For live flight data:	Live flight data	Flight record	Ground test
Plus voice message when a critical level reached and no **EscC** available	Battery voltage **EscV**	Maximum rpm **EscR+**	Rpm **EscR**
Plus voice message when at 30% mAh	mAh used **EscC**	Maximum current **EscA+**	Current **EscA**
Plus voice message every 10 seconds	Altitude **Alt**	Maximum power **Watt+**	Power **Watt**

The natures of the data are shown in standard weight letters. The names of the data on my systems are shown in bold. I fly mode 2 so it is safer to move my left hand to read the live flight data. They are in any case read out to me. Where a datum is not available such as altitude on an aerobatic power model the box will be left blank.

GPS sensor

Just plug it in and let it see the sky. These are the data that the GPS sensor produces:

GAlt	Altitude from the GPS sensor (not zeroed for field altitude)
GSpd	Ground speed from the GPS sensor
GPS	Map reference from the GPS sensor (more precise data on telemetry screen)
Hdg	GPS heading

Logical switches

A logical switch (L01 to L32) would have been better called a ‘virtual switch’. It is a test of some sort which can then trigger an action. For example you might read the temperature of the motor. When it reaches a danger value the logical switch causes a warning to be read.

Variometer

There are two versions of the vario: ordinary and high precision. Each uses an air pressure sensor to detect climb or descent and makes use of the built-in **Vario** and **Val** software. **Vario** generates tones and **Val** allows a value to be spoken. For the X8R Rx you can unplug the leads for Data in and Data out. You only need to use the three pin SmartPort lead.To zero the sensor for your field's altitude, and the day's air pressure, put the model on the ground, open the telemetry screen, hold down **ENT** and click **Reset Telemetry.** You could programme a switch to do it. I pondered over ambient pressure changes and vario height readings. I reasoned that pressure goes up as you go lower. If the vario tells me I am lower than the ground then the pressure must have gone up. QED or is it?

These are the data that vario sensor produces

Alt	Altitude	(must be zeroed by Reset on each flight session)
Vspd	Vertical speed	(may also be calculated from GPS data)

The standard vario tones are:

Ascent:	on-off tone	Both: High rate – high pitch Low rate - low pitch
Descent:	steady tone	

This will be the setup:
The three position switch **SD** will select:

Centre:	quiet
Up:	Play tones using 'Vario' function
Down:	Play the value of Alt every 10 seconds

A logical switch will be set to say when the maximum height has been reached (122m/400ft if that applies to your flying field).
Make sure that the vario is connected and detected or the data and options will not appear.

On the **Logical switches** screen set:

L01	**a>x**	**Alt**	**122m**	(switches on when Alt more than 122m)
		(a)	**(x)**	

On the **Special functions** screen set:

SF1	**SD↑**	**Vario**	
SF2	**SD↓**	**PlayValue Alt 10s**	(says altitude every ten seconds)
SF3	**L01**	**PlayValue Alt 10s**	(looks at logical switch L1)

For the last SF hold down **ENT** and select **Logical Switches**
If you want to change the pitch of the vario tones move to **RADIO SETUP**.
Here are special functions to reset telemetry using switch SE and to play a sound

SF4	**SE-**	**Reset**	**Telemetry**	□
SF5	**SE-**	**Play Sound**	**ratata**	**1x**

If you want to switch off sink tones and just have rising ones, set the **low dead band** to **OFF**.

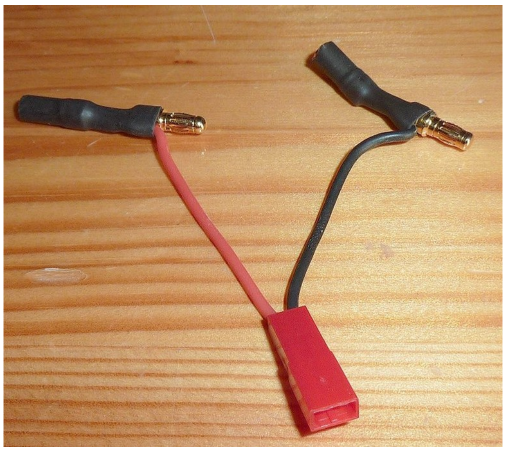
Setting default switch positions

When you switch on the Tx it checks that the switches are in their correct default positions. You want switch SD in its central position by default but will get an error message unless you change its default position.

Move to **MODEL SETUP**
Move to Switch Position and set SD to its central (blank) position.

RPM/temperature sensor

The RPM sensor needs a wire tapped off from each of any two of the motor leads. This is the version that is used in the test bed, which I made from a JST male extension lead. Each wire is soldered to a male and a female banana (bullet) connector, which are soldered back to back with a thick wire. Heat shrink holds it secure.



Some setup is needed for the RPM part of this sensor.
First you need the number of coil sets on the motor.
You can get this from the data sheet or you can count them.
In this case there are six coil sets.

Setup is done on the sensor itself.
There are four tiny LEDs and a button labelled **Key**
Hold down **Key** until the first LED lights
Then press down **Key** repeatedly until you get the correct pattern for the pair number.

This is the table supplied with the sensor. For a 6-set motor it is **Off/On/On/Off**:

Pairs->		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
LEDs	0	*		*		*		*		*		*		*		*
	1		*	*			*	*			*	*			*	*
	2				*	*	*	*					*	*	*	*
	3								*	*	*	*	*	*	*	*

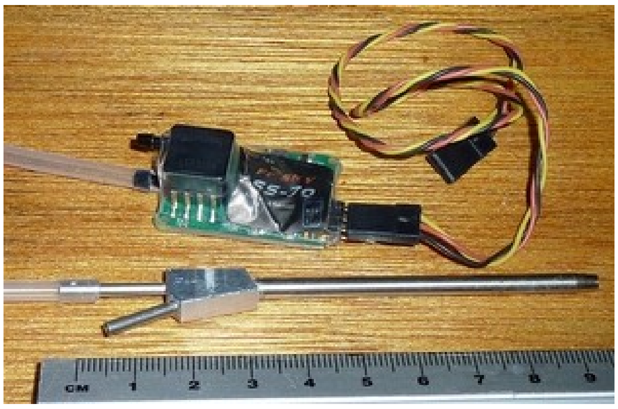
* = LED LIT

You can see that it is the binary code for the number of sets. There are 10 types of people. Those who understand binary and those who don't. Don’t also set the number of coils in the edit page for this sensor. Either do one or the other. I do it in the sensor edit page.

Airspeed indicator

FrSky makes two. The cheaper one, shown below, measures airspeeds up to 270kph (168mph). The other, at twice the price, goes up to 360kph (224mph). I can’t really see the point of the latter. The first reads speeds more than enough for normal flyers and the second is no use for the dynamic soarers that now go way above 360.

The device is small and light. It looks very similar to the pitot tubes used on full-size aircraft so should be suitable for all but the strictest scale models. As you would expect it must be placed in free airstream away from propellor wash. Outboard on one wing or in the fin seem obvious choices. The manual recommends that the side holes in the pitot tube are **at least** 13mm from the surface it projects from. Two light plastic pipes go from the tubes to the sensor. One metre of 2.2 mm bore pipe is supplied.



Only one pipe is shown here. I use 2 mm pipe as well.

After connecting up, blow a hairdryer or heat gun, set on **cool,** at the front of the two tubes.

This shows the pitot head. You can just see the holes in the side that provide the second (ambient) pressure reading.

The data item from the sensor is called **Aspd.** If you move to the telemetry page you should see the sensor detected and the data displayed and changing. If not, select **Discover new sensors.** You can then add the datum to your telemetry screen. Change the unit from knots, if you want, by selecting **Edit** on the line for the sensor.

Here is the ASI fitted into the canopy of an ageing Bixler for test flights. As you see it is very compact.

The central tube in the pitot head is just a straight-through tube. If it becomes blocked you can clear it by pushing a steel wire through from the rear.

Testing went very well. Obviously the Bixler didn't go very fast. The sensor reacted very quickly and showed readings of 12 knots on glide and around 35 knots under full power level flight. Why knots? It was at the request of the person I was training. For models, like high aspect-ratio tapered-wing gliders, with a tendency to tip stall at low speed near the ground, feedback about airspeed could be very useful. There was of course no way to test the accuracy as a GPS sensor gives ground not airspeed. Maybe I'll try on a dead calm day, if we ever get one.

Current sensors

FrSky sells two current sensors, 40A and 150A. I hope they introduce one in between. 40A is a bit low for anything other than small and medium models. 150A is over the top for all but large and EDF models, and it has to be threaded on to the red ESC wire so requiring unsoldering the connector. It's probably best to make up a special lead with XT90 connectors on each end. The 40A one has male and female XT60 connectors as you see from the picture. Alternatively you could use Neuron ESCs. Details further down.

OpenTx has a very clever telemetry function called Consumpt. This takes the current (Curr) from a 40A FrSky current sensor, and presumably time from its clock, and calculates how many mAh of energy have been used. It removes the guesswork involved in relying on the varying voltages from a lipo sensor. However only the 40A version reads the battery voltage as wel, called **VFas**. For the 150A you would need to have a lipo sensor to give the data item **Cels**, if you want a voltage reading.

You can calibrate both **Curr** and **VFas**. This involves using a multimeter to read the true current and voltage and comparing them with the values shown in telemetry. If either is different, you open the **Edit** screen for the datum and enter a **Ratio** to correct the difference. For example the telemetry shows 22.5V and the meter shows 23.6V. The ratio is 23.6/22.5 or 1.05.

Setting up consumpt

With the current sensor and the battery connected:
Move to **TELEMETRY** screen
Look for **VFas** and **Curr** in the list of data
If they are not there move to **Discover new sensors** and press **ENT**
The data should now appear.
(If they don't, check that the sensor LED is flashing. If not check connections.)
Move to **Add a new sensor**
Click **ENT**
Type in a suitable name. I used **mAh**
Change type to **Calculated**
Change **Formula** to **Consumpt**
Change **Sensor** to **Curr**
EXIT
The new sensor **mAh** should appear in the data list.
You can test it out now without having to add it to a telemetry display screen.
Run the motor and watch the number rise.
Now you can add it to your telemetry screen.

Testing the accuracy of Consumpt

I did a bench test with a tethered model. Unlike the nitro heads I was able to do the tests indoors in the warm, not at 3°C in my workshop. I used a brand new, newly charged, 2200 mAh Zippy Compact lipo and a similar, used, newly charged, Graphene Panther. Each had similar measured internal resistances of about 3mΩ. I ran the motor mostly at about half throttle until the Consumpt reading showed that 75%, 50% and 25% of the energy was left. I then stopped the motor and noted the battery % readings by connecting a battery capacity meter to the balance lead, though of course **its** accuracy is not certain. The meter was disconnected whilst running as its consumption would not be metered by Consumpt. I also noted the battery voltages when stopped. The results were:

		Zippy		**Panther**		
% left	**Consumpt**	**Meter %**	**Voltage**	**Meter %**	**Voltage**	**'Standard' data: lipo volt v %**
100	0	100	12.60	100	12.62	12.60
75	550	74	11.82	74	11.85	12.07
50	1100	44	11.45	46	11.47	11.55
25	1650	14	11.15	21	11.20	11.03
Volts on chargers after test			11.23		11.20	

For the record the full-throttle current was 37A on the Zippy and 39A on the Panther. That

was nearing the limit for the device and it was only a Tundra motor running on 3S with a 12x6 propellor. However the person who wrote the OpenTx code seems to imply that the sensor will take higher currents.
The results are pretty consistent but show a lower %age than that calculated by Consumpt especially towards the end of the 'flight'. I checked my battery meter against another to see if it might be reading incorrectly. It does give a lower % reading than my iSDT charger when I connect the batteries after a flight, though slighly higher than the second meter. We must remember that none of these devices are calibrated test instruments. They are much too cheap for that. They are probably 10% tolerance. What the tests show is that this method of monitoring batteries is reasonably accurate, and will help prevent battery damage and dead-stick landings.

I have now carried out flight tests using the 40A sensor and consumpt. I have found it easy to use and accurate. I land at about 1550 mAh used and measure about 20% left. When recharging, my charger shows an amount of mAh being put back that is very close to that measured as used by consumpt.

150A sensor

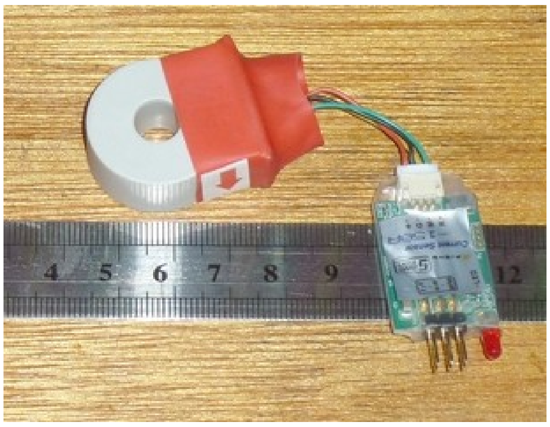
It is a little larger and heavier than the 40A but not much. It threads on to the red lead from the battery to the ESC. The arrow must point **from** the battery. Remember current is conventional current from **+** to **-**, rather than electron flow.

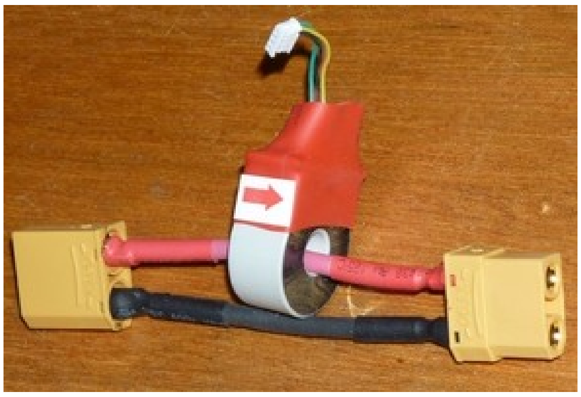
I'd love to know how the sensor works. It looks like a mains current clamp meter, but they only work on alternating currents as they measure induced current. From battery to ESC it is direct current. Perhaps it is a Hall Effect transducer which works with steady magnetic fields. If so why the coil? Or does the coil create a steady field that the varying direct current changes? Anyone know?

I have now had a chance to test out the current sensor in a Goldwing Slick model. This uses two 4500mAh batteries in series. I opted for 3500 as the safety limit. On my last flight I landed with about 3000 showing as used. This means that the batteries should be down to 33%. When I tested both after the last flight, the meter showed that each was 30%. So once again FrSky telemetry proves accurate, with a confidence in this case of plus or minus five percent or better.

Here are pictures of the sensor and an XT90 12 awg lead I made up for it.

Sensor





Lead

Reading out the state of the battery

There is more than one way, apart from the display on your telemetry screen. The simplest would be to read out the mAh used when energy reaches a low level. Set a **Logical switch** to switch on when mAh reaches 1500 in the above test case. That will switch on a **Special function** that reads out the value every 10 seconds. You would need to enter a suitable value into the Logical switch code line for each model's battery size.

Here is the code for that:

L01	**a>x**	**mAh**	**1500mAh**		Switches when battery getting low
SF1	**L01**	**Play Value**	**mAh**	**10s**	Speaks mAh value every 10s after 1500 reached

Any L or SF number can be used.

Or you could use a message. For example you could use **L01** to speak out **batlow** ('battery low') when the consumption hits 1500. See elsewhere for how to copy the message into your active list. Or you could read out the mAh every twenty seconds.

Data logs

Press **ENT** on one of the data in the sensor list and select **Edit**. You can then move to **Logs** and press **ENT** to switch it on. This data will be logged to the SD card for later viewing on your computer. A new file is opened each day when used. The data are in CSV format so they can be saved and edited in LibreOffice Calc, or Microsoft Excel if you are slumming it.

One use of data logs is to record GPS position data, perhaps with a logical switch to start it when speed is greater than zero or the model is above the ground. You can then use Companion to send the data to Google Earth and superimpose the flight path onto the map. This can be done not only from above but from side view as well.

Trainer: Setting up a buddy box

Start with the master Tx. Move into **MODEL SETUP** and move to **Trainer Port Mode**
It's probably already set to **Master/Jack**. If not then set it to that. This makes the Tx the master. You will need a slave. In theory the Taranis should buddy with any Tx with the standard 3.5mm jack socket. This flexibility means that the setup is a little more complicated. Using another Taranis as slave is very simple to set up.

Move to **Radio setup/TRAINER**
You see a screen something like this:
TRAINER

	Mode		%	Source
Thr	**:=**		**100**	**CH1**
Ail	**:=**		**100**	**CH3**
Ele	**:=**		**100**	**CH2**
Rud	**:=**		**100**	**CH4**
Multiplier	**1.0**			
Cal	**0.0**	**0.0**	**0.0**	**0.0**

The first column shows the control on the master Tx - the one controlling the model.
The second is what happens when the trainer switch is moved: **:=** mean 'Replace'

The third is what percentage of the slave control throw is transferred to the master
The fourth is the source channel on the slave that replaces the control on the master
The bottom line has four numbers.
Connect master and slave using a mono 3.5mm jack plug lead in the left-hand socket on the back.
Switch on the master and the slave.
Moving the slave sticks will alter the numbers on the bottom line.
Centre all slave sticks **including the throttle (**line up the markings)
Move to **Cal** and select for edit.
Press **ENT** .
All of the four numbers should go to about zero.
If using an electric motor, for safety, set to ‘engine off’.

Move the slave sticks about and check:
    Are the slave sticks controlling the correct master channels?
        If not change the **Sources** until they do
    Are the slave sticks producing roughly -100 to +100 on all channels?
        If more than 100 then reduce the **%ages** in column three
        If any are less than 100 then increase the **Multiplier**
        Adjust the %ages for each stick to get close to 100.
    Do the slave sticks give the same direction of servo travel as the master?
        If not then scroll to set the % value to **-** (**minus)** the value for that stick
Now we need to select a switch and write **Global functions** to transfer control. Although it is on the wrong side for mode 2 trainers, use switch **SH** as it is the only two-way momentary switch on the Taranis.

Move to **Global functions** screen
Using the usual methods set the data until it looks like this (choose the next unused GFs)

GF1	**SH↓**	**Trainer**	**–-**	**☑**
GF2	**SH↓**	**Play Track**	**trnon**	**1X**
GF3	**SH↑**	**Play Track**	**trnoff**	**1X**

Try handing over control using switch **SH**.
Switching might cause the servos to twitch or move. If it does then try correcting it with trim on the slave. If not then go back to the start and try again.

Recording a screenshot

You might want a picture of your Taranis’ screen to store a speed or height record, or perhaps a last moment GPS location when you see that your model is about to crash. It is easy to create a special function to store the screenshot on the memory as a .bmp file. Ideally you will want to use SH, the only momentary switch. You will need to disable its use as a trainer switch or instant trim.

Here is the code:
SF1 **SH↓** **Screenshot** To see the screenshot you will need to copy it into a computer.

Flight mode

Suppose you have a model that can be flown in different ways. If you want to fly slowly you might have some flap and a little down elevator trim to compensate. You might want to have more flap and more trim to land or take off. For aerobatics you will want everything set neutral. Each of these is a **Flight mode**.
Move to **MODEL SETUP/FLIGHT MODES and y**ou see a screen like this:

FM0			**:0**	**:0**	**:0**	**:0**	**0.0**	**0.0**
FM1		**–-**	**:0**	**:0**	**:0**	**:0**	**0.0**	**0.0**
FM2 etc								

FM0 is the default flight mode

The first empty column allows to add a name.
The second **–-** allows you to select a switch and position.
The next four are trim settings for **Rudder, Elevator, Throttle, Aileron.**
The last two are how quickly the changes happen called **Fade In** and **Fade out.**
Change them in the usual way.
You might also want to have voice messages in **Special functions** as you move the switch. There are twenty-two to choose from.

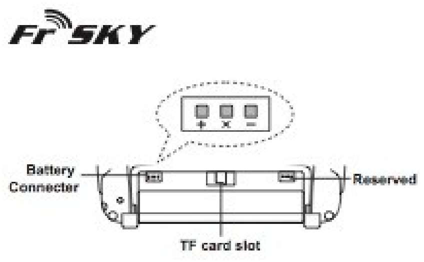
Instant trim

When trimming a new model you sometimes have to hold sticks off centre to keep the model flying stably. If you set a switch to give instant trim, it will add the stick displacement(s) to the trim. This will reduce the servo throw in the direction of the trim but it will enable you to land safely and make proper adjustments. If you are not using **SH** for trainer you could use it for this, though it is the wrong side for mode 2 flyers. It is best to disable the special function by unticking it after trimming your model. Hitting it by mistake in normal flight will probably be disastrous. I think it is best to play with it on the ground before using it in flight.

This is how the special function will look if using switch SB, both centre and down positions:

SF10	**SB-**	**Inst. Trim**	**☑**	(could be any SF number)
SF11	**SB↓**	**Inst. Trim**	**☑**	

Transmitter batteries



The Taranis operates when the Tx battery supplies between 6 and 15 volts. It is currently sold with a 2000 mAh NiMH battery pack that replaced the original 800 mAh one. This is said to give about six hours of run time. As shown in the table below, 6.5 V is the lowest safe voltage. The Tx voltage is shown on the main screen and you can set an alarm to sound at 6.5.

Calibrating the Tx voltage
It is a good idea to calibrate the screen battery voltage reading. Remove the battery and measure its voltage with a digital voltmeter. Replace it and move to page 7 of **RADIO SETUP** (**ANALOG INPUTS**). Look at **Battery Calib**. Set it to be the same as the voltage you measured rounded to the nearest 0.1V. It probably will be out by no more than 0.1V but it pays to be sure.

The supplied NiMH batteries are charged in place in the Tx using the plug-in power supply. All other batteries, such as Lipo or LiFe, must be removed from the Tx for charging. The main board connector is small and possibly fragile so it is best to stick with the NiMH or to use an adaptor lead for the others that means you don’t have to plug into the main board each time.

On current models the charge light blinks when charging and goes off when the battery is full. On older model transmitters the charge light was solid when charging.
If you change battery types, update the voltage range and warning values on page 1 of the **RADIO SETUP** screen as shown in the table below.

Battery	**High**	**Low**	**Warn**	**Notes**
NiMH 800 mAh	8.0	6.5	6.5	stock battery on older model
NiMH 2000 mAh	8.0	6.5	6.5	stock battery on newer model
2S Lipo	8.4	6.6	6.9	can plug into main board plug
3S LiPo	12.6	9.9	10.2	need an adaptor lead
3S LiFe	10.8	8.4	9.9	need an adaptor lead

Setting up a low voltage warning

The "Battery Meter Range" setting field on page 1 of **RADIO SETUP** only changes the battery icon display. The low battery warning must be set in the "Alarms/Battery Low" field for the appropriate voltage applicable to the battery size/chemistry you are using.

Updating FrSky X8R receiver (Rx) firmware

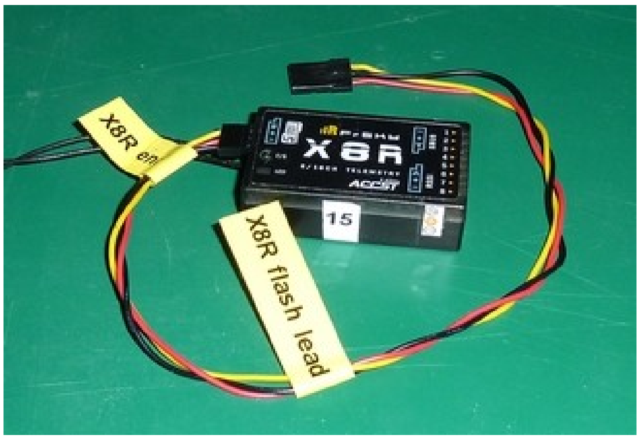
You might need to do this if your receivers are old and need updating or are imported and need to have the EU **Listen Before Talk (LBT)** firmware installed. Your X9D transmitter (Tx) will need to have version 2.1.0 or later of its software for this to work. You will need a proper computer with a memory card slot to use these instructions. If you have a toy tablet or similar you will have to research how to get the software file onto the SD card, perhaps with a plug-in card reader. It can also be done by using a USB connection and OpenTx Companion, again needing your research.

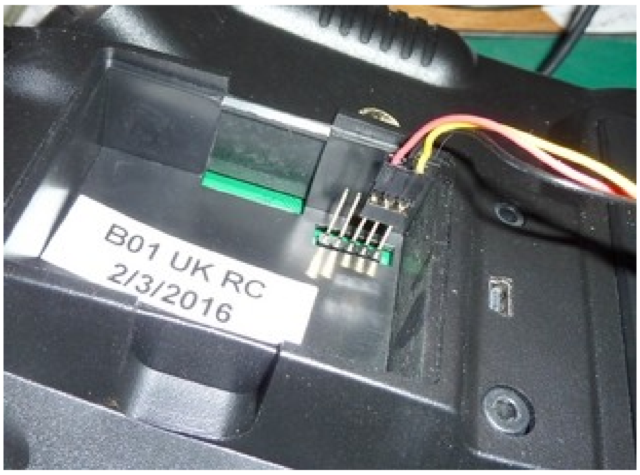
NOTE: The filename formats only apply to Europe where Listen Before Talk (LBT) is required. Elsewhere in the world this will not apply but it should not be difficult to work out which file applies to you.

Make sure your Tx battery is fully charged. Warning: The Tx pins provide the full voltage of the Tx battery. Use the standard NiMH Taranis battery whilst you update

the devices. If you have changed to a lipo or other battery you risk damaging your receiver and telemetry devices due to the higher voltage.

1 Make up a lead to connect the X8R to the pins in the module bay of the X9D transmitter. The + and – leads (usually red and black) are reversed at the Tx end as shown. The best approach is to convert a thick double-female servo-type lead. You might be able to push the two connectors out of the black shell to swap them but you will probably need to cut the wires and crimp or solder new female connectors on and use a new shell. Make sure you label the lead so you don’t use it for anything else and blow it up.





2 Download the file from https://www.frsky-rc.com/x8r/. This will be compressed in zip form with a name like **X8R X6R FW20170210.zip**. If you double click this name, your computer’s operating system should convert this automatically. There will now be a folder with the same name as the file. Open the folder, and possibly sub-folder, and you will find a file called something like **X8R_EU_build7022102.frk**. This is the software you need to install (flash) in your Rx.
3 Remove the SD card from the battery bay of your Tx.
4 Insert it into the card slot on your computer.
5 Open the card and move to the **FIRMWARES** folder.
6 Copy the **X8R_EU_build7022102.frk** file to this folder.
7 Return the card to the Tx.
8 Open the module cavity on the Tx.
9 With the Tx switched off, plug the lead into the Tx as shown in the picture, and into the

SmartPort slot on the Rx.
10 Switch on the Tx and move to **RADIO SETUP**.
11 Move to the **FIRMWARES** folder.
12 Click the .frk filename.
13 From the drop down menu click **Flash External Device**.
14 You see a progress bar and the Rx alternates red and green LEDs.
15 When the progress bar completes switch off the Tx and unplug.

Hopefully your Rx will now be upgraded and will bind successfully.
This method can also be used to update the firmware on other devices such as telemetry.
It should work on any receiver in the X series that has a SmartPort for telemetry.

Updating the transmitter OpenTx firmware

Again if you don't have a proper computer you will need to search the web or your manual to find out how to copy the files onto your SD card. Firmware is the geeky name for software that is permanently stored in a silicon chip, though it can be overwritten. It is safer for control systems as it is very unlikely to be corrupted.

Turn your transmitter off and remove the SD memory card (it's in the battery bay).
Put it into a carrier then put it into the card reader in your computer.
This is a good time to make a backup of the existing card contents to your computer. This is especially important for models.

Now get the new contents for the card.
Go to https://www.frsky-rc.com/taranis-x9d-plus-2/ then select **FIRMWARE**
Download the correct software which will almost certainly be at the top of the list.
For EU mode 2 flyers currently this is **FW-X9DP-V2.2-20180525.zip**.
Find the file and double click it.
It will create a folder containing various versions.
Open this folder and look for a file similar to **x9dpEU-mode2-20180525.bin** (the numeric date element might be different)
Copy and paste this into the **FIRMWARES** folder on the SD card.
NOTE: It is probably a good idea also to download the file for the firmware you are currently using, just in case the new software causes problems.

Eject the SD card and put it back into the transmitter.
Push the rudder and aileron trim switches in towards the centre, and switch on the transmitter.
The screen will display **Taranis Bootloader**
Choose **Write Firmware** and then press the **ENT** button.
Click the correct file and press **ENT**.
Wait till activity stops then switch off.
Switch on again and select **RADIO SETUP**/page 5.
Check that the **VERS:** number is now the new one.
When you switch on you might get **SD CARD WARNING. Expected version 2.2V0016** or similar. This means that the files on the SD card, especially the sound files, are out of date.
If you don't get this message the update is complete.

If you get the message:
On your computer go to https://downloads.open-tx.org/2.2/nightly/sdcard/

Select and download **sdcard-taranis-x9-2.2V0016.zip** or whichever version you have been asked for. 16 now appears to be unavailable so choose the highest number.
Double click it to unzip it and open the folder into which the files have been put.
Open the **MODELS** folder.
It will be empty.
Copy all of your models to **MODELS** from whatever backup you have made.
Open the **SOUNDS** folder
Delete all of the folders except **en** (unless you use a different language from English)

Remove the SD card from the transmitter.
Insert it into your computer
Delete all of its contents
Copy all of the folders from your computer to the SD card.
Eject it.
Put it back into your transmitter.
When you switch on again you will not get the warning message and the sounds will work.

ACCST V2

In 2022 FrSky receivers started to be sold with ACCST V2. This fixed a couple of bugs that no-one I know has ever experienced. The upshot is that to bind to such receivers your transmitter firmware must be updated. If you do that then you must update all of your older receivers as well. This is not a good move by FrSky. If you are about to buy more receivers make sure your supplier is willing to change the software to V1.

Using the S.BUS interface

One of the benefits of using FrSky is that the X8R receiver comes with an S.BUS socket. This is an abbreviation of ‘Serial Bus’. S.BUS enables you to have one wire feeding an area of your model that might have many servos and other devices. For example a wing might have servos for aileron and flap, an ESC (multi engines) and a retract. That means four signal wires. With S.BUS you only need one. S.BUS also allows you to use all sixteen channels in the X8R. Normally you need two receivers, one set to channels 1 to 8 and the other 9 to 16.

Using standard servos with the S.BUS to PWM Decoder

You do not need to buy expensive S.BUS servos and other devices. FrSky sells a small 15g device called an “S.BUS to PWM Decoder”, that plugs into the S.BUS receiver socket through a servo lead extension and then connects to four standard digital or analogue servos. PWM is Pulse Width Modulation, which means that the receiver sends a series of pulses to the servo. A short 1ms pulse moves the servo to one end of its travel and a long 2ms one to the other end.

People report that analogue servos sometimes don’t work with the decoder. I tested three analogue servos. Of these the Hextronic HXT900 failed and the Tower Pro MG90S and Turnigy TGY-1440A worked. I have found that some retracts don’t work. There seems to be no pattern about which will.

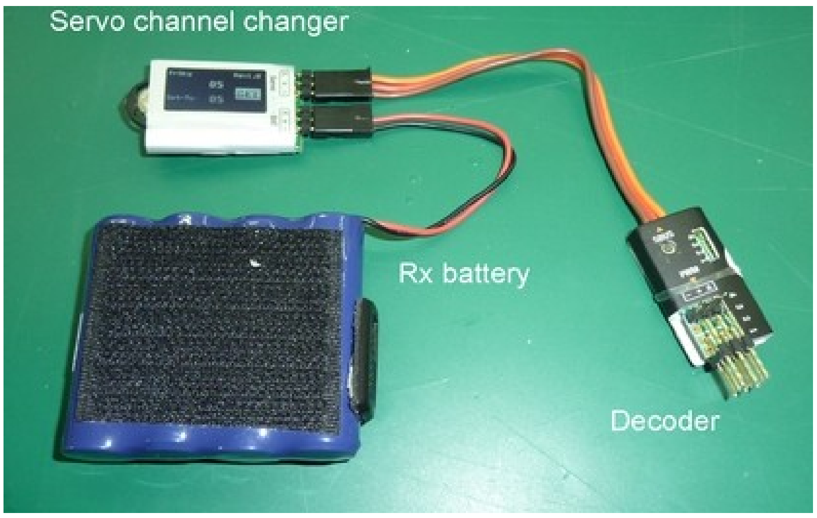
You need one decoder in each wing and perhaps a third in the fuselage to operate rudder, elevator and tailwheel devices. For three decoders you need two Y-leads at the receiver end or perhaps you might make a three-way. See my comments about wire sizes below.

On a normal receiver you assign a channel to a servo by plugging it into the correct channel on the receiver. On S.BUS all devices are on one branching wire. Each device has to be given and store a channel number. For this reason you do not need to label the leads and connectors. To set the numbers for non S.BUS servos you use a device, called a “Servo Channel Changer”, to set the channel numbers in the decoder.

Channel changer

Setting channels on the decoder

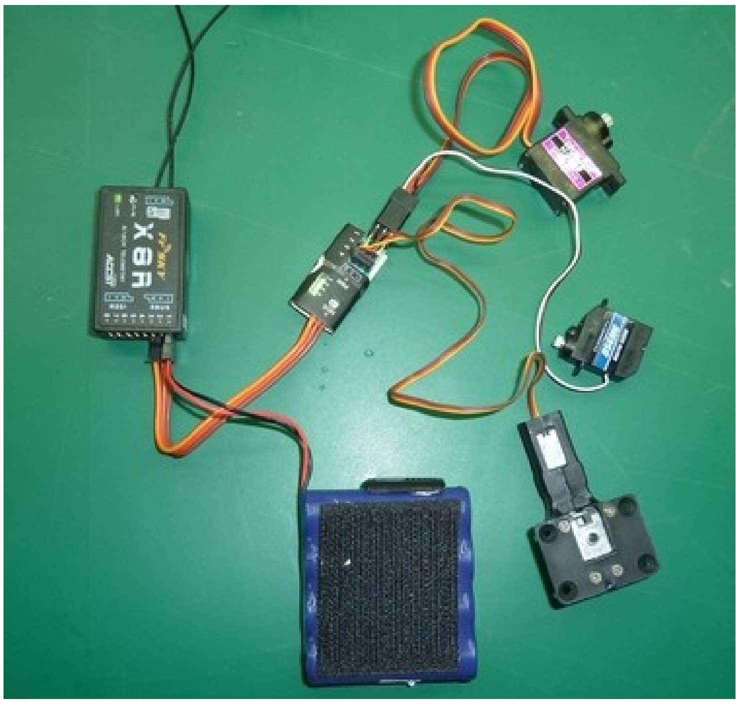
Plug the **S.BUS** lead from the Decoder onto the **Servo** pins on the Servo Channel Changer (**SCC**) Connect a receiver battery to the **BAT** pins on the **SCC**. This is shown in the picture.



You will now see the SCC screen light up, together with one LED on the decoder.
This LED shows the channel you are changing, probably 1.
The upper of the two numbers on the SCC shows its current channel setting, if there is one.
Now you have to use the SCC thumb wheel which is a bit fiddly.
In general you push it in to select or save something and push it clockwise (CW) or anti-clockwise (ACW) to change something.
The display will probably show **SET** lit up.
Push ACW and the lower number will be lit up.
Push in the wheel.
You can then change the value by CW or ACW pushes.
When you get the channel you want push in.
Push CW to light up **SET.**
Push in the wheel.
You see the upper stored value change.
The first time is the worst.
After that you will know how to do it.

Now set channels for the other pins.
Using a soft point push in the recessed button next to **SBUS** on the decoder.
You see the LED for another servo channel light up.
Go through the above for this and the other two sets of servo pins.

A typical setup for one wing



I decided to try out S.BUS, possibly to go in the Mosquito I am building. This is the set-up on my bench with an analogue and a digital servo and a small retract, but without the ESC connected. It all worked perfectly. Note the single wire from the Rx.You use a Y-lead to connect another decoder. That is the benefit of S.BUS.

Mixing S.BUS and conventional channels

You can continue to use the conventional receiver channels, even one that is also used for S.BUS. For example you could use channel 7 in S.BUS for retracts in the wings and plug the tailwheel retract direct into the Rx channel 7.

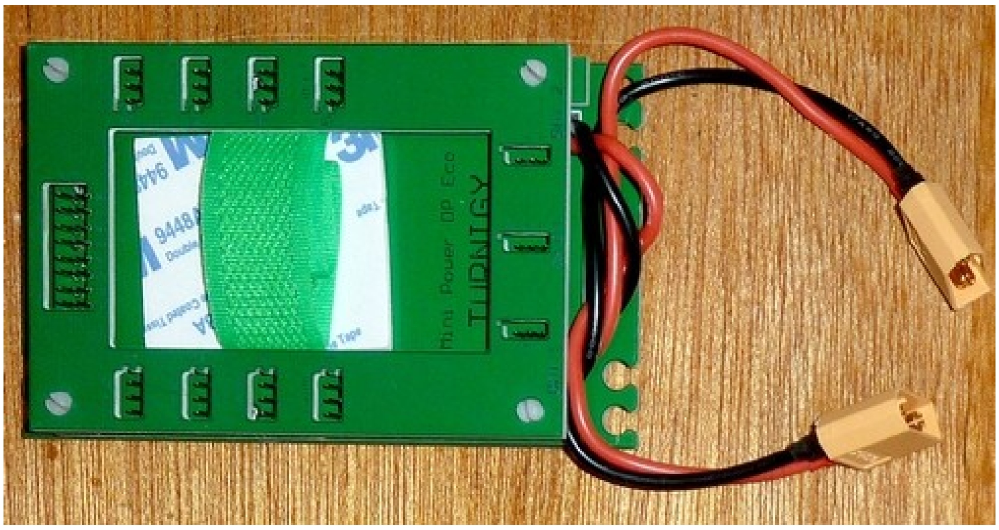
Currents and wire sizes

There is one other consideration. If planning to use high-torque servos, or large retracts, the sizes of the wires will probably need to be increased, especially the ones from the decoders to the receiver. Remember that the currents for all devices travel down one wire. It would help to use high voltage servos (6 or 7.4V) as this will reduce the current for the same servo power. The wire that plugs into the S.BUS socket on the receiver will need to be the largest. The three sizes that are usually used for servo leads are 26 awg (flat), or 24 or 22 awg (usually twisted). These safely carry about 1, 3 and 5 A respectively. These numbers are very uncertain. Every table you look at gives very different values because it depends on cooling and the material of the sheath. Digital servos often draw a heavy current when starting to move or when stalled. By the way I am sorry to be using American Wire Gauge (awg) instead of modern metric, but that is how servo leads are specified.

Here are some typical values (surprisingly difficult to discover):

Retract unit	500 mA (or more)
Servos for aileron and flap	500 mA each (for a 15 kgcm servo)
Throttle lead for ESC	Unknown but probably low mA

Apparently some electric retracts for very heavy models can use 5 or 6 A, which is presumably why pneumatic retracts are used. Some high-torque servos can draw 3 A or more. However most appear to be around the figures shown. So it looks likely that for ordinary large size models 26 awg will be fine for each wing but perhaps when the wires all join up 22 or 24 awg would be advisable. An alternative, which is the best way if your model has the space, is to use a power distribution board (below). This passes on the servo signals but provides higher current directly to the servos etc.



Planning a FrSky S.BUS installation

Setting up S.BUS is different from conventional radio. There is only one connection to the Rx so each device, such as servos and retracts, has to be given and to store a channel number. FrSky uses decoders so you have to plan which receiver channel is allocated to each of the four connections on each decoder. I used it first on a Mosquito. In each wing there were aileron and flap servos, together with a retract and an ESC. In the fuselage there were the usual elevator and rudder channels as well as a separate servo for tailwheel steering and a tailwheel retract. I will have two different Rx throttle channels so I can balance the motor powers exactly. So I decided on three decoders, one in each wing and the other in the fuselage. I also considered a sound system for the twin Merlins, which would require a fourth decoder, possibly the single channel FrSky one.

I was forced to devise a table to help with the planning and have added this to the end of the manual with another filled in with data from my Mosquito.

The new frequency band – 900 MHz

Radio control flyers have a new frequency, nominally called 900 MHz, but in fact 868 MHz in the EU (don’t they always love to complicate things). The lower frequency and higher powers give a greater range – up to 10km - and the longer waves will diffract (bend) better round obstructions. 2.4 GHz has 12.5 cm waves and the 900 MHz 33.3 cm. At present only FrSky is offering 900 MHz kit in the UK. However the manuals are very brief and lack essential information, so I was left to find things out myself with some valuable help from RCLife and RobotBirds.

I only found out about the new band when I bought a new Taranis transmitter. As a special offer it came with a free external R9M module to fit into the module bay in the back. I think this frequency is the result of lobbying by the drone industry. 2.4GHz probably doesn’t give enough range for commercial drone use.

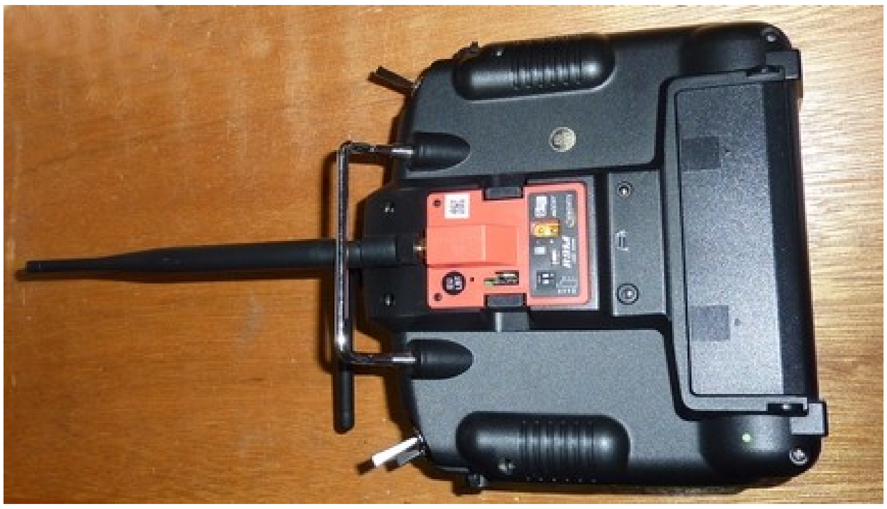
Three transmitting powers are available on 868: 25, 200 and 500 mW. This compares with the Taranis 2.4 power of 60 mW, though now you can legally use up to 100mW. However if you want to use telemetry only the 25 mW power can be used. Again I assume this is a pointless EU intrusion. It is also more evidence that the frequency is for drones, as a distant, possibly commercial, drone needing the higher powers would have its own return channel for data. So what starts out looking like a bonus for us from drones turns out to be less revolutionary if we want telemetry, at least in the EU. However there are still benefits.

I decided to buy a receiver to try it all out. I got the full-size R9 model with the usual range of servo, S.BUS and SmartPort telemetry ports. It is about the same size as the X8R receivers that I usually use and weighs 16g, but is rather more expensive at £46.95. Binding and custom failsafe is as easy and as good as on the X8R. The two aerials are floppy on the stubs and of course longer. There are two other receivers of reducing size and weight with the lightest being just over 1g. Manuals for all of these are elsewhere on my site.

R9M module

R9 receiver

In the Taranis



What you do on the transmitter

The first step is to ensure that your transmitter has the latest OpenTx software. At the time of writing this is V2.2.2 dated 25 May 2018 though V2.2.1 will work. With V1.6 the R9M module will not show up in the transmitter’s list. There is an explanation in this manual about how to update your Tx firmware.

The next steps are to update the firmware on the R9M module and the receiver. You do this as described above, except that you must ensure that you update the **EXTERNAL** module not the internal one. The files you currently need (though the date numbers might be different) are:

R9M module:	**R9M_LBT_20180428.frk**
R9 receiver:	**R9_LBT_20180425.frk**

One symptom of out-of-date firmware is that the servos do not move smoothly but jump in larger steps, so it is best to do the updates at the start.

On the Taranis you can have both the internal and an external module active, though not at the same time. You select which you want to use for each model, not for all models. I removed the cover on the module bay and pushed in the R9M module. You must stick to the standard NiMH internal battery, or at most a 2S lipo, as a 3S lipo will give too high a voltage. The aerial screws onto a standard radio frequency SMA connector and looks a little weak. Best to take a lot of care not to strain it. I remove mine to go in the Tx box. Suppliers like Farnell stock a wide range of 868MHz aerials if you fancy a change or even more aerial gain.

On the back of the module there is an XT30 port for a 2S battery to be plugged in. You will need to change your battery connector to female XT30. I wondered why this was needed and then realised that in 0.5W mode the internal battery would run down very quickly. The normal seven hours might become under one hour. The module takes its power from whichever battery (internal and external) has the higher voltage. The two DIP switches are for setting up PPM mode, but we will use R9M mode. The manual says that the Tx white button is a ‘Function button’, but the only possible reference to it is that it might be the ‘Failsafe button’ for PPM mode.

Select a model.
Go into **Model Setup**/page 2
Move down to **Internal RF**
Edit **Mode** to read **OFF**
Move to **External RF**
Edit **Mode** to read **R9M** and **LBT(EU)**
Then bind the receiver in the usual way.
On Ver2.2.2 you have to select which mode to use during binding, in this case:
25mW Ch1 – 8 Telem ON.

Despite choosing Ch1 – 8 there is no reason to think that S.BUS cannot be used to give 16 channels. However I haven't had a chance to try that out yet.

System testing

Once I had updated the transmitter, module and receiver firmware the receiver bound in the usual smooth and quick Taranis way. You can just replace an existing X8R with an R9 with no changes needed. I plugged it all up and it worked, including telemetry. Prior to updating the firmware the servos behaved in a jerky manner, with response delays. If you change mode, say from 25 to 500 mW you have to rebind.

Flight testing

I installed it in a trusty old slow flying Bixler for flight testing. I displayed the Rx signal (RSSI) and the vario height on the Tx screen so I could check its range and telemetry. I expected that the lower 25mW radiated power would give at least as good a range as the 2.4GHz 60mW. The result was much better than expected. I flew the 1.5m model to what I would normally think of as a visual distance limit, probably about 1km, and the RSSI was still in the mid 90s. When flying closer it stayed on 100.

Drawbacks

One drawback is the 200mm length of the receiver aerials. One can easily be laid horizontally in most fuselages but the second one is more difficult and there is no firm guidance as to best routes. Apparently it doesn't have to be at right angles and the Rx might work with only one. On the Bixler I taped the second one to the leading edge of a wing but it could be poked into holes in the wing root ribs in future models. Another drawback is the possible need for an external battery if the internal one runs down too quickly in high power mode. Even with a flat form 2S lipo it means using a transmitter tray.

How many simultaneous users?

The R9M module is labelled ACCST which is also the protocol used for 2.4GHz. The latter uses the Frequency-Hopping Spread Spectrum (FHSS) system. Provided there isn't a lot of external interference on the 2.4 GHz band used, having ten or more simultaneous flyers seems to be safe. Unless the system used on 900MHz isn't normal ACCST there should be no difference from 2.4. If you know better please let me know. One slightly disconcerting fact is the the 868MHz band is also used for such things as remotes for garage door openers, though they will be of very limited power and range.

Conclusion

I wondered how I might use this new kit. When might any of us need it? If you are flying FPV with a spotter, using either a drone or fixed wing, you might need the higher power. If you are a visual fixed wing flyer you are unlikely to need the greater range. So the only time it might be useful is if the 2.4 GHz band becomes congested or your field suffers interference from other types of 2.4 transmission. Or of course you might just want to show off how modern you are. What me? The aerial (No! Not ‘antenna’, as I am only half American) is longer and distinctive so it will spark curiosity. Overall I am very impressed. Once again, ‘well done’ to FrSky.

A legal word of warning is needed about the 900 MHz band. Contrary to some rumours, it is not illegal to use the band in the UK though if you cause interference you might get into trouble. This will just be a warning from Ofcom and then you must stop using it. You won’t be ‘shown the rack’ in the Tower. The Tx transmits at 25 mW by default. Only at the 500 mW setting might you cause a problem. The other possible problem is if you fly close to mobile phone masts. These will transmit at 900 so your model might experience interference, though the frequency hopping, FHSS and encryption from Code Division Multiple Access should protect you.

Using a Taranis transmitter with a model flight simulator

I checked whether the Taranis would work with my flight sim, Phoenix. It had to be switched on to be sensed by the software. Then it calibrated and worked perfectly. I await reports from others who are trying it with other sims, including Real Flight.

Linking a Taranis wirelessly (not Phoenix)

The word ‘wirelessly’ is a bit misleading. In fact you use a USB dongle that is a receiver. The 2.4GHz control signals are sent from the Tx and picked up by the dongle. The Tx and dongle must be bound in the usual way for an Rx. FrSky makes a dongle as does Orange. The Orange is cheaper but can only handle six channels so installation has a few more steps to do with the number of channels. Here I will describe the FrSky one. There are videos on the web for the extra bits for Orange.

Plug the dongle in and wait for the HID driver to be installed (Human Interface Device)
Create a new model, best named after the simulator you are using
Make sure channels are set to D16
Set the Tx into bind mode (not close to the dongle)
Remove the dongle
Hold the dongle bind button in
Plug the dongle into the computer
After a short while the red light will flash and the green will be steady showing it is bound
Release the button
Remove the dongle
Exit the Tx from bind mode
Plug the dongle in again
The dongle should now show steady green
The Tx should now connect with the sim.
What happens now will depend on what sim you are using. You are on your own. I use Phoenix and this will not connect with it as it requires the wired USB box with the licence in.

Using the X9D with a flight simulator

This will not tell you how to set up for a particular simulator (RealFlight etc). It will tell you how to change the channels to match what the sim expects. Let the model sit on the runway, then move one stick at a time to see what each stick movement does to the model. That way you will be able to see what needs changing.

Create a new model, calling it after the sim you are using.
Move to the MIXER page using PAGE.
To change a channel:
Use the – button to move to the channel you want to change, for example Ele.
Hold down ENT.
Press ENT to select Edit.
Move down to Source using the – button.
Press ENT to select it for edit. It will flash.
Move the stick you want for this channel, for example Rud.
The characters will change to the required stick in this case Rud.
Press ENT to fix it to the new value. It stops flashing.
Press EXIT twice.
Using the + or – button, move to the next channel you want to assign.

My thanks to Keith Eldred for that.

Using your Taranis with PicaSim

PicaSim is an excellent model flight simulator for Windows and for Android devices, written by Danny Chapman (“Mr Rowl”). It can be made to run on Apple using PlayOnMac though I can’t describe that as I don’t use a Mac machine. The software is free but, very reasonably, Danny requests a donation to help the project so for everyone’s sake please do donate when you are happy you like the software. Download it from http://www.rowlhouse.co.uk/PicaSim/download.html.

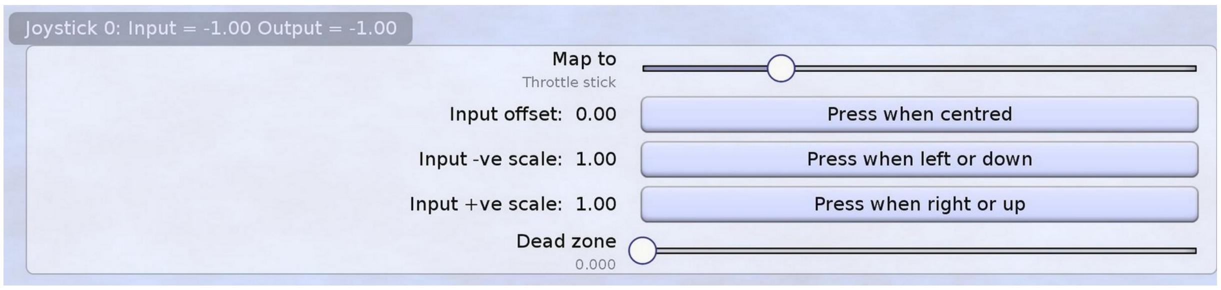
To use your Taranis as the controller, simply get a longish lead – at least 2 m - to connect the USB port on the back of the transmitter to a USB port on your computer or other device. Note that the Taranis USB is not a microUSB.

To get you going I will summarise the basics very briefly:

Switch on the Tx. You must do this before plugging it into the computer.
Plug in the Tx.
Start up PicaSim and click the **cog wheel** for options.

Click into **Joystick**.
Tick **Enable joystick**.

Here's what you see as you scroll down for the first stick movement, in this case probably the throttle stick:


Joystick 0: Input = -1.00 Output = -1.00

Map to
Throttle stick

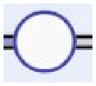
Input offset: 0.00 Press when centred

Input -ve scale: 1.00 Press when left or down

Input +ve scale: 1.00 Press when right or up

Dead zone
0.000

Waggle the throttle stick.
You should see the **Input** and **Output** numbers change.
If they don't scroll down and find which numbers they do move.
Click and drag the slider and set **Map to** to **Throttle stick**.

This is the slider 

After setting to the correct movement you have to calibrate the movements:

- Centre the stick
- Click **Press when centred**
- Do it for **left or down** and **right or up**

Do this for the remaining three stick movements - pitch (elevator), yaw (rudder), roll (aileron)

To reverse a joystick scroll far down and select **Invert** for the correct stick.
Switch on throttle stick as brakes if you are going to slope soar.
Save with a suitable name, e.g. Taranis.

You can now choose **Aeroplane** and **Scenery**.

Click **Back** to the flight screen.
Start a flight.
Waggle the sticks and see the left and right displays move.

Other transmitters. I don't have any knowledge of Txs that don't have a USB port. There are leads available that plug into the trainer port. You will need to find out from forums and youtube. Anyway this is a manual for FrSky.

Curves: a practical example

Setting up a Taranis X9D for the Hobby King 2.4 m paramotor

What the paramotor needs

A paramotor is a wing-type parachute sail. The pilot has an engine strapped to his or her back. There are two controls, called brake lines or just brakes. One is held in each hand. To turn left, the left brake is pulled. This pulls down the left side of the sail and increases the drag on the left causing a left turn. When landing, both brakes are pulled to cause a stall just above the ground. Control of height is done entirely by the throttle.

In many ways the setup is a bit like a V-tail. The difference is that the servo movements are in only one direction.

Each servo is at maximum when the aileron and elevator sticks are central. Both servo arms are high. As the aileron stick is moved left, only the left servo arm moves down. The right does not move. Vice versa for the aileron stick moving right. When the elevator stick is pulled back both servo arms move down.

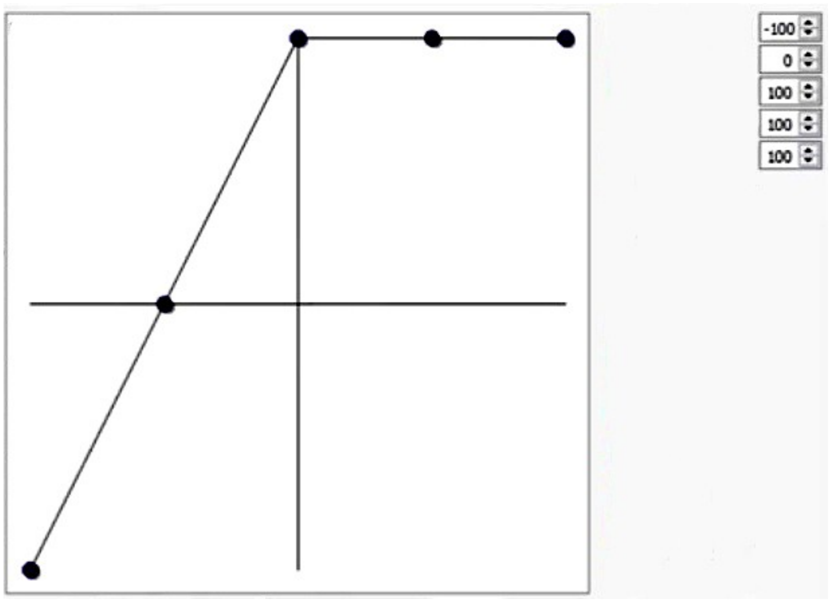
This is what is needed:

- Half the movement of the sticks must be ignored.
- The half that is not ignored must give full servo travel.
- There must be brake line trim that applies equally to both servos.
- There must be brake line trim that varies one servo compared with the other.

This requires some clever mixing and the use of curves. I do love the Taranis. Normally you think what you want to do and then do it using the three screens : INPUTS, MIXER and OUTPUTS. However in this case it was more complicated. I’ll cover the curves first.

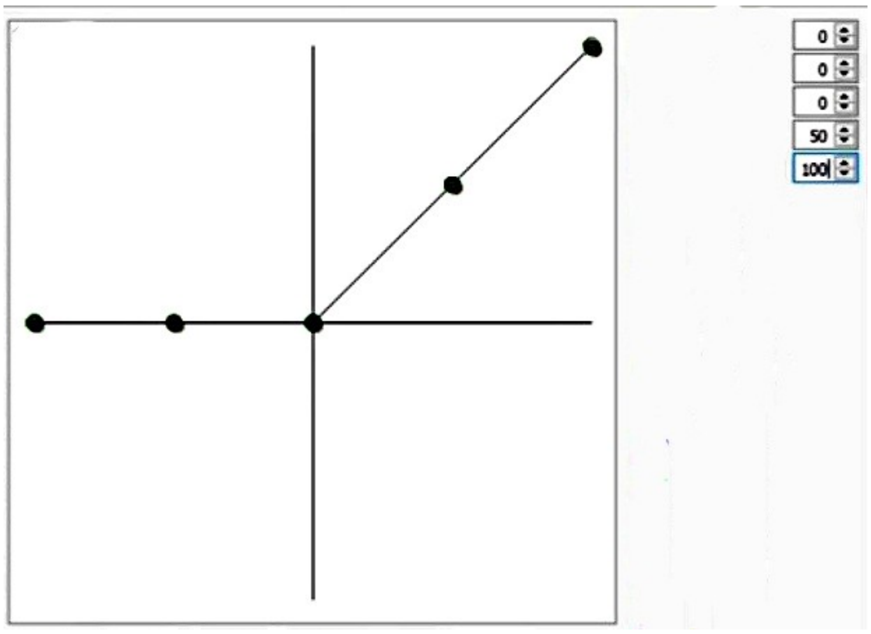
CURVES

Elevator Curve 1   Name: Ele

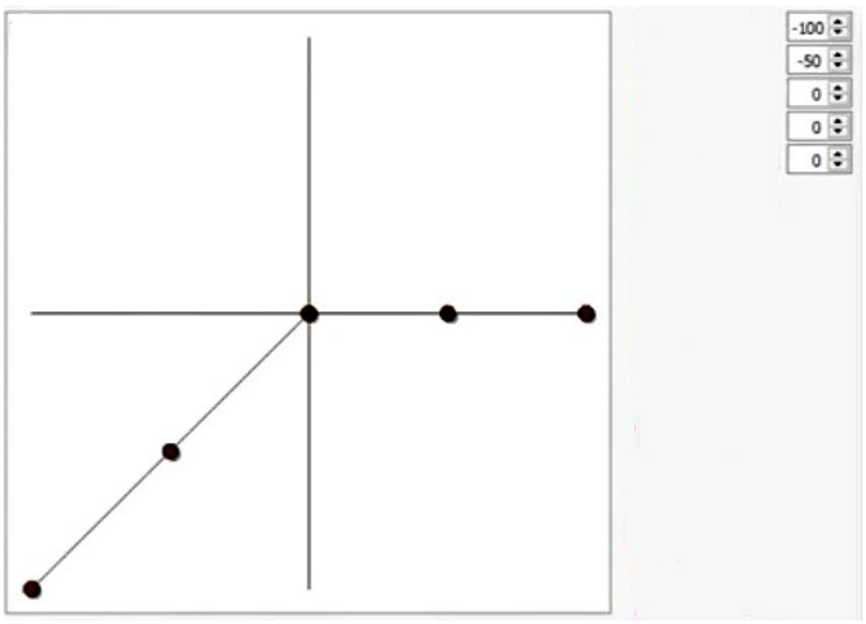


-100
0
100
100
100

Curve 2 **Name: CV2**


0
0
0
50
100

Curve 3 **Name: CV3**


-100
-50
0
0
0

Now the rest

INPUTS

Thr	100	□	Thr		
Ail	100	□	Ail	CV2	
Ele	100	□	Ele		
Rud	100	□	Rud		(not used)
Ail2	100	□	Ail	CV3	
Trim	100	□	S1	CV3	(or your choice of rotary)

MIXER

Ch1	100	**I**	Thr		
CH2	100	**I**	Ele	Ele	LBrk
+=	200	**I**	Ail2		
+=	50	**I**	Trim		
CH3	-100	**I**	Ele	Ele	RBrk
+=	200	**I**	Ail		
+=	-50	**I**	Trim		

The aileron weights are 200 as only half the stick movement is detected.

OUTPUTS

In **MODEL SETUP**

Tick ‘**Extended limits**’
Then set throw to 125%

CH2	-125.0	125.0

Special functions and logical switches

If you are using a lipo voltage sensor these special functions will warn you when you hit a low voltage. These will also allow you to set engine off with a switch.

SF1	SC-	OverrideCH1	-100	☐	Throttle cut
SF2	SC↓	OverrideCH1	-100	☐	
SF3	SC-	Play Track	engoff	1X	Says ‘engine off’
SF4	SC↓	Play Track	engoff	1X	
SF5	L01	Play Value	Cels	10s	Says lipo volts every 10s below 10.8 V
L01	a<x	Cels	10.8 V		Switches when battery getting low

Other settings

Don’t forget to set failsafe to throttle zero and everything neutral.
A suitable picture for your screen is Paraglider.

Testing your transmitter aerial

The Taranis has a built-in aerial tester. This checks that all of the signal is getting though the feed wire from the circuit board and out to the aerial. It is similar to, though not the same as, the Standing Wave Ratio that radio technicians use to set up their aerials for maximum efficiency. In the Taranis it is called Relative Antenna Status (RAS).

To test the RAS of your Taranis:
Switch on
Hold down **MENU** to go to **RADIO SETUP**
Press **PAGE** to get to page 7 **ANALOGS TEST**
At the bottom of the page you see **RAS.**
Hopefully its value will be 1 or less.
If it isn't there might be a problem with your aerial.
Up to 2 is probably OK.

Grasp the aerial with your hand.
The value will go higher, perhaps to 16 or more.
Don't hold it for too long as the energy is now going back into the transmitter and might harm it. The risk is small as one of my Txs has had a high RAS value for some time due to a fault. It came to no harm. Repairing it is the next section.

Repairing or replacing your transmitter aerial

Though these notes are about the FrSky Taranis X9D the general principles should apply to many makes and models of transmitter.

If, when you switch on, you get the message 'WARNING Tx antenna problem' then you probably have a faulty aerial (antenna). Even if the model appears to respond normally do not fly until it is fixed. Your range will be lower and you won't know by how much. Use the transmitter aerial test described in the last section to check the RAS. You can also use the test to check whether the repair has worked or whether part of the electronics has also failed.

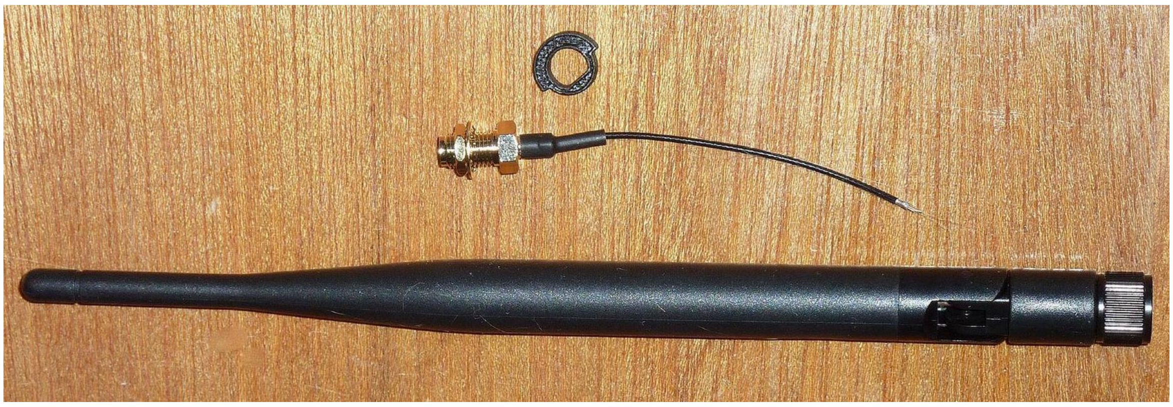
The standard folding X9D transmitter aerial is fairly fragile. Unless you have the version of the board with the plug in IPEX connector on the board, replacement requires some delicate soldering. Don't attempt it unless you have a steady hand and a good quality iron with a very fine tip, ideally one with temperature control. Otherwise sweet talk the club electronics expert. He or she will usually respond to flattery.

If you are replacing the aerial with another standard one you can skip the next bit and move straight to 'Soldering in the new feed wire.'

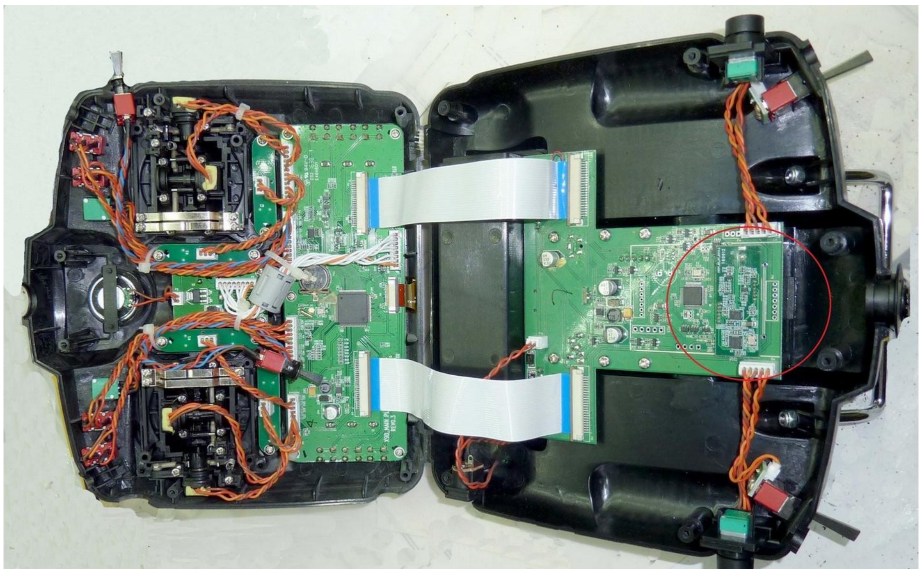
I decided to replace the standard aerial with a removable one that screws on. For this I had to install a different fitting called an RP-SMA connector. I bought a kit from banggood for about £7. If I had only wanted the aerial I would have gone to t9hobbysport.com.

The aerial is detachable, though the connectors only have a life expectancy of five hundred times screwing on and off. You can also use the RAS to check if the RP-SMA needs to be replaced after it has been used a lot. There is a small loss of signal compared with a direct soldered aerial connection. Most connectors drop the signal by 1 dB or so but I bought a 5 dB aerial rather than the standard 2 dB so that should compensate. The extra 3 dB in theory doubles the power. The aerial is longer so I will have to remove it for storing in the box. The higher gain aerial is more directional so don't point it at the model when flying.

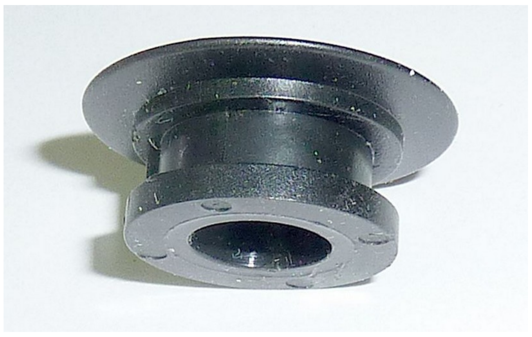
The RP-SMA fitting and 5 dB aerial kit. The 2 dB one is a bit over half the length.



After removing the six screws the case falls open. The part we are going to deal with is in the red circle. It’s a pity that the tiny board can’t be removed from the larger one behind it. It would make the work on it much easier.

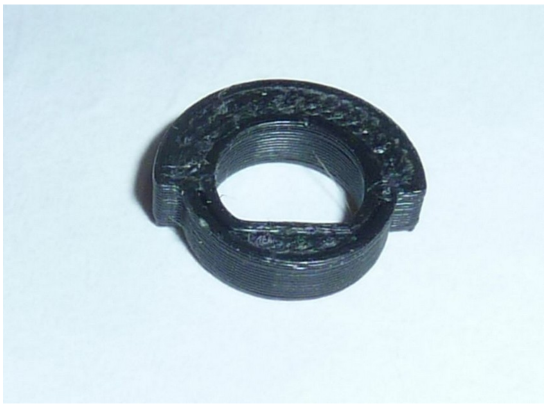


I had to trim the plastic bush in the top of the Tx so the threaded part of the RP-SMA fitting stuck out far enough. With a fine hacksaw and a craft knife I removed one layer of the top and was left with this.

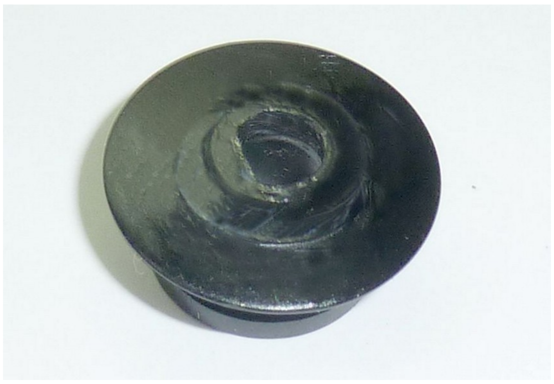


I then glued the plastic washer that came with the new aerial into the bush using epoxy. Notice there is a flat that locks the connector so it can’t rotate. You will probably need to open out the central hole in the rear of bush so the hexagonal part of the RP-SMA connector fits in. I used a 10 mm drill bit, handheld and not in a drill, and then made grooves for the points on the hexagonal body with a burr in a Dremel.

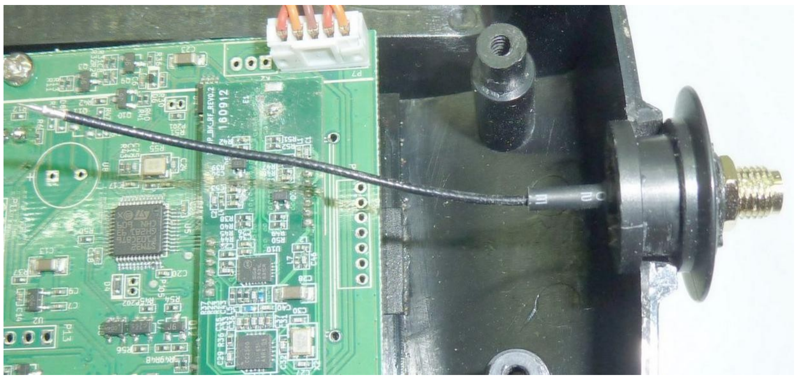
Washer

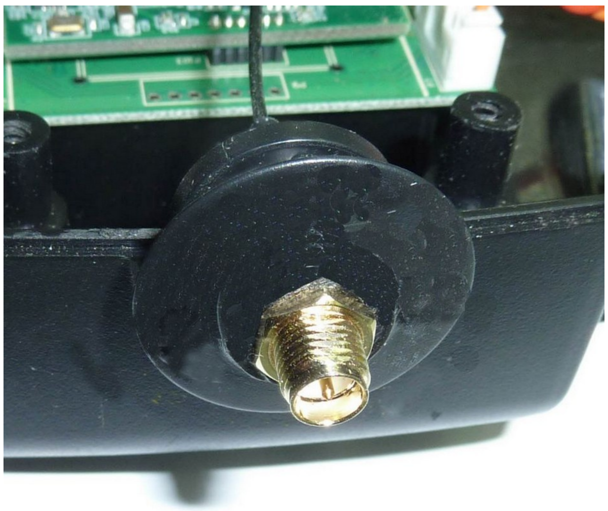


Washer glued in place in bush



These pictures show the bush fitted into Tx before soldering. Fixing the bush first reduces the strain on the newly soldered joints later. I added masking tape to lock it in place.

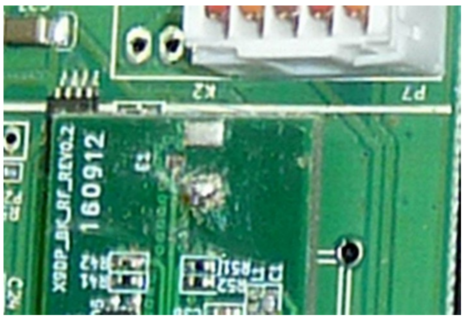


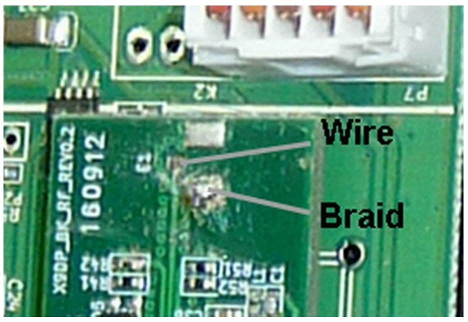


Soldering in the new feed wire

If you have jumped straight here take a look back at the picture of the Tx opened up and the red circle. Unscrew the six screws and open the case. Take a photo of the solder connections especially if your Taranis is a slightly different model from that in the pictures. It will also help with switch orientation as some will pop out of their housings.

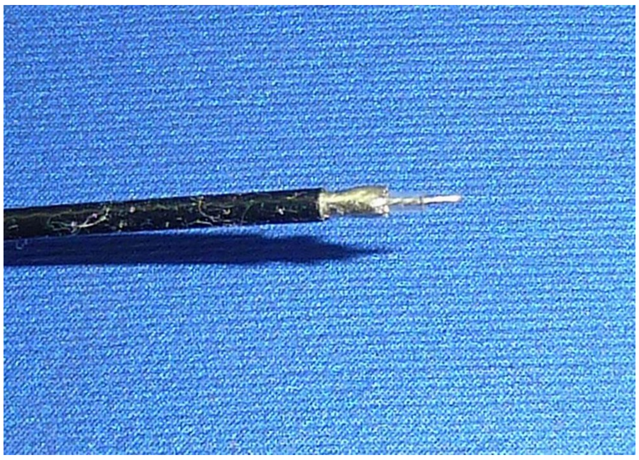
Using the soldering iron, remove the old feed wire from the board. Clean up the solder pads using a solder suction tool or braid, taking care not to overheat the board. The two pads will look something like this. The lower blobby one is what the braid solders to. The small dark pad up and to the left is where the core wire goes. In my case that pad had pulled off so I had to scrape the green varnish off the track leading to the pad with a scalpel and solder to that. Keith Eldred suggested carefully checking the pads and track. If there is any sign of them becoming detached then refix with epoxy. It will withstand brief heating when soldering.





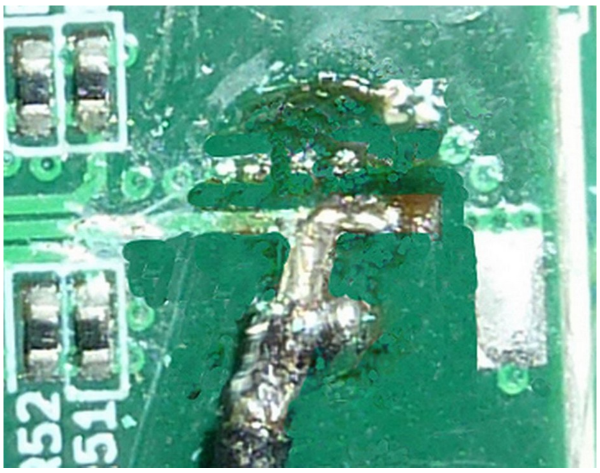
I found a bench magnifier essential.

The next picture is the new feed wire from the RP-SMA connector as supplied. There will be a similar one if you are replacing without using the RP-SMA. You can see the bare end of the wire on the right, then the plastic insulation, then the outer braid and finally the outer black sheath. The bare end must be soldered onto the left hand pad and the braid to the lower one. You don’t need lots of solder but make sure it is shiny and rounded when hardened. A grey or cloudy surface means a poor, ‘dry’ joint, which must be done again.



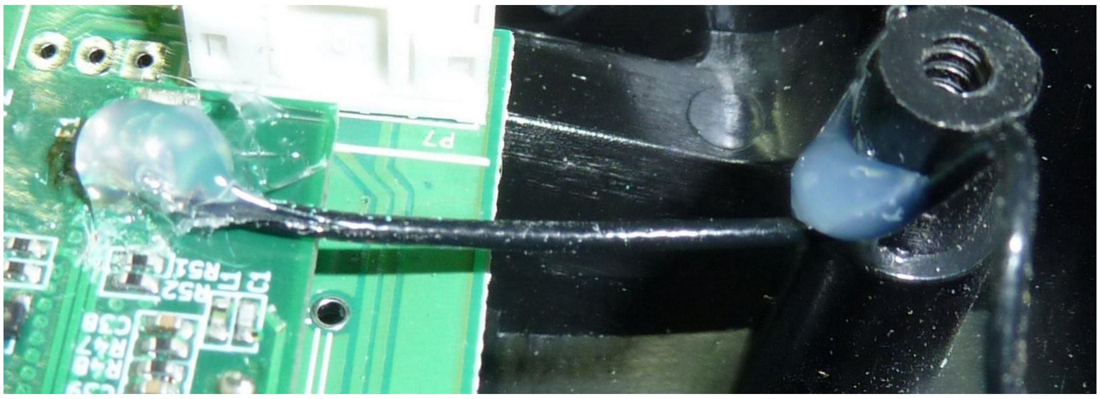
The braid is the tricky bit. Heat the braid in advance and put a blob of solder on without melting the plastic insulation. A touch of the iron on the bare wire will freshen up the tinning.

Here is my first attempt - the most tricky soldering I have done for years. I had to file down my thinnest tipped soldering iron bit to match the tiny areas to be soldered.



First I scraped the track again with a scalpel and tinned it. Then I scraped a bit further back baring some more track to allow a continuity check with a meter. I bent the bare wire end round to match the direction of the track. Then I soldered the braid and finally soldered the bare wire taking just a few seconds for each. The continuity check showed that both were connected and not shorting. I couldn’t find my macro lens so the picture is a bit blurred and I have tidied the picture up a bit with Photoshop for clarity.

Finally, I locked the wire onto the board by putting on a blob of hot melt glue using a glue gun. Do not forget this step. It is crucial to ensure that the wire is firmly fixed in place so I also glued it to a pillar as you see in the next photo.



Oscar Liang suggests using hot glue to fix the feed wire down onto the case of the transmitter. This is only possible if the feed wire is fairly long and mine wasn't. He also suggests locking the RP-SMA connector into the plastic bush and case. I didn't need to do that as there was a notch to lock it and I had cut grooves for the hexagonal body. His website OscarLiang.com is an excellent source of technical information.

You will know if the repair has been successful when you switch on. If it isn't, you will get the error message again. In my case the RAS was 1 and when I wrapped my hand around the aerial it shot up exactly as it should.

Dummy load in place of an aerial

The Tx that needed repair was one that I mostly use as the slave on a training buddy connection. I don't need the aerial on it for that, especially the new longer 5 dB one. An aerial feed wire can safely be disconnected from an aerial provided there is a load connected to it to soak up the power and stop it being reflected back to the transmitter. This is called something like a '50 ohm RP-SMA dummy termination load'. The power produced by the Tx is low at less than 100 mW so any SMA dummy should do as the minimum power they can take appears to be 500 mW.

The replacement aerial and RP-SMA connector are unusual in that the aerial is female, having a metal socket, and the RP-SMA is male having a metal pin. Usually it is the other way round. By the way please don't get wokey about the use of male and female, and the word mate meaning to fit them together. Note that they apply to the metal parts – pin or socket – not the cases. They have always been the words to use in electronics and unless new ones are invented we are stuck with them. Any new words are likely to have ten times the syllables anyway. How about Pin socket inter connect handed output PSICHO. Couldn't think of a Y word.

You have to very careful to buy the correct gender of dummy load. It must be female, that is having a metal socket. Weirdly these are often described as 'male' but be guided by the seller's pictures.

IPEX connectors

There is another way to do this repair. You can buy IPEX male connectors that you can solder to the board. Some receivers use them for their aerial wires. I haven't tried this. They are tiny and very difficult to work with. You would then need to fit a different RP-SMA connector with the feed wire fitted with a mating IPEX socket connector. I think some Taranis Txs were supplied with these connectors at one time and the layout of the pads on the circuit board appear to match the layout of the tabs on the connectors.

However the connectors are very small and designed to be soldered to the board using a wave tank rather than manually. However if you are good at soldering and want to be able to change RP-SMA connectors more often it might be worth a try. The mated connectors would have to be locked using hot melt glue.

Male IPEX connector to be soldered on the board or connected in some way.

Socket female IPEX connector on the feed wire

Dummy resistor

I realised after doing all that work that if I was only ever going to use the Tx as a buddy slave it didn't need an aerial at all. I could have got away with just soldering a 50 ohm resistor across the two pads. Oh well, another working Tx might be useful one day. Maybe when I drop my present main transmitter.

Repairing an X8R/RX8R receiver aerial

In a nutshell you can't repair one, but they are cheap and easy to replace if damaged. To open the Rx case push a small flat screwdriver gently into each of the four catches. Two are under the connector polarity stickers. The catches will click and the case will come apart. Look at where the aerials connect to the board. It is almost certain to be an IPEX connector described above. Twist the damaged aerial wire off. It just pulls off.

You need to buy a new aerial exactly like the other one. Your FrSky dealer will advise. If you can't get an identical one it is probably best to replace both. See whether the old glue on the board will stop the new connector going fully on. It probably won't but you might need to trim it - carefully. Press the new connector on firmly then lock it in place with a blob of heat gun glue. To reassemble the case first find the half case with the slots for the S.BUS and RSSI ports. Slide the board in so the pins go into the two slots. Locate the aerial wires in the cutouts then press the halves together till they click. Job done. It goes without saying that you must do thorough function and range checks before flying the repaired Rx.

Pages in each navigation route

Route 1: Current model settings

Home page:	Displays one model
Page 2:	Displays stick, trim, rotary and toggle switch settings
Channels monitor:	Bar charts for current control positions

Route 2: Radio setup pages

Radio setup:	Date, time, sound volume, backlight, units, mode
SD card:	See what files you have on the card
Global functions:	Detect events and data values to trigger actions **on all models**
Trainer:	Set up buddy box
Version:	Check you are using latest version
Switch test:	Allows you to test that switches and trim controls are working
Analog inputs:	See the values being input from your sticks and other controls
Hardware:	Don't understand this
Calibration:	Twirl the sticks and knobs to store maximum values

Route 3: Model setup pages

Model setup:	Name, image, bind, range, timers, failsafe, RF module etc.
Heli setup:	Whirlybird stuff – a mystery to me
Flight modes:	Set trim for different flying styles
Inputs:	Control sticks
Mixer:	Links inputs (e.g. sticks) with outputs (e.g. servos)
Outputs:	Servo setup including rate, offset, and reverse
Curves:	Graphical display of rate, offset, reverse and expo
Global variables:	A value you want to use in several places
Logical switches:	Detect events and data values to trigger actions **on one model**
Special functions:	Selectable actions triggered by switches and other controls
Telemetry:	Detect sensors and set up display pages

An attempt to explain Lua

Lua (proounced ‘Lew – ah’) is a programming language. The code, which is similar in some ways to C, is called a **LUA script.** They can be added to the SD card in the Tx and used in functions. It was developed in the 1990's in Brazil, and the name comes from the portuguese for 'moon'. It allows extra flexibility to some functions, for example how telemetry values are combined to give calculated results. It is not used for any function that might cause a model to crash if the script stops working. It is most unlikely that you will ever need to use it. For more information look at: https://en.wikipedia.org/wiki/Lua_(programming_language). On a BMFA webinar I learned that Jeti transmitters make use of Lua, mostly for data analysis and screen displays. The company has a set of ready written scripts that you can download. If you like to code take a look at http://www.lua.org/. Be warned. Installing it requires computer knowledge.

List of functions

Global functions

Trainer	Allows two transmitters to be connected for training new flyers
InstTrim	For early flights when stick are held away from neutral. Sets subtrim.
Reset	Resets selected data like timers, telemetry data etc.
Set timer	Set a timer to a value
Volume (sound)	Sets how loud the transmitter sounds are played
SetFailsafe	Sets control surface and throttle settings if signal is lost
PlaySound	Plays a sound a programmable number of times
PlayTrack	Speaks a recorded voice track a programmable number of times
PlayValue	Speaks the value of a variable with programmable frequency
LuaScript	Runs a Lua script
BgMusic	Background music on transmitter presumably for freestylers
BgMusic II	Background music on transmitter
Vario	Allows data to trigger actions like play altitude or tones for climb
Haptic	Sets up sensory feedback like vibration
SD Logs	Logs telemetry data to the SD card at specified intervals
Backlight	Turns backlight on
Screenshot	Capture a picture of the current screen and store to memory

Special functions

All global functions plus:

Override	ChX	(Channel X)	Overwrites the channel with a specified value
Adjust	GVX	(global variable X)	Set the global variable to a specified value

Glossary for OpenTx and FrSky

2P	A lipo battery that has cells arranged in two parallel groups
35C	A lipo that provides current 35 times the capacity in Ah. Other values.
3S	A lipo battery of **3** cells in **S**eries. Also 1S to 12S
ACCST	Advanced Continuous Channel Shifting Technology. See FHSS
Ah	Capacity of a battery in amp-hours. See also mAh
Airbrake	Device that projects from a wing or fuselage to slow the model
Airspeed indicator	Telemetry sensor that gives true airspeed
ASI	See Airspeed indicator
Audacity	Free software to edit sounds on linux, Windows or Apple computers
Backup	Making a copy of a model or file to be stored in a different place
BEC	Battery elimination circuit e.g. a 5V output from an ESC
Binding	Linking a receiver and transmitter excluding all others
Buddy box	Linking the model's transmitter to another to allow training
C rating	Measure of how much current a battery can supply. Multiply the capacity in Ah by the C rating, e.g. capacity 3Ah, C 25, current 75A.
Capacity	Energy capacity of a battery or cell in amp-hours (Ah) or more commonly milliamp-hours (mAh). Also Ah is maximum charge current.
Channel	Socket in the receiver that connects with a servo, ESC, gear etc

Consumpt	Telemetry battery datum that tells you how many mAh you have used
Crow brake	Powerful airbrake produced by down flaps and up ailerons
Curve	Graphical image that shows stick versus servo movement
Custom failsafe	Setting up positions for all controls in the case of loss of Tx signal
dB	Decibel. Way of comparing two signals, sounds or voltages
Detent	A position into which a control clicks and is held gently
Differential	Having an aileron move more up than down
Dual rate	Increasing or decreasing servo throws using a switch
Dummy load	50 ohm device that screws on to SMA connectors to absorb signal
Engine	Internal combustion (IC) power source for model
ESC	Electronic speed controller for electric motors and power to receivers
Expo	Number 0 – 100 that shows how gentle the servo throw is near to zero
F/S	See Custom failsafe
Failsafe	See Custom failsafe
Feedwire	Coaxial 50 ohm wire that connects the Tx board to the aerial
FHSS	Frequency hopping spread spectrum. Transmission method used in FrSky transmitters where it is called ACCST
Firmware	Software that is permanently stored in a silicon chip, though it can be overwritten. It is safer for control systems as it is very unlikely to fail.
Flaperon	Using ailerons to move down together to produce flaps but also roll
Flight mode	Allows you to switch to different trim for different styles of flying
FrSky	('freesky') Chinese company that makes radio control equipment
Global function	An action that applies to all models stored on the transmitter
Global variable	A stored value that can be used by any function
GPS	Global positioning by satellite data sent by receiver through telemetry
Haptic	Sensory feedback to pilot, usually a vibration
IC	Internal combustion engine: diesel, glow, petrol or gas turbine
Input	Control signal from a transmitter stick, switch or rotary control
Instant trim	Adds stick position to trim during initial test flights
IPEX	Miniature coaxial connector for radio signals
Lipo	Lithium polymer battery used for powering electric flight motors. See also 3S, 35C, mAh, 2P
Lipo sensor	Telemetry sensor that measures flight battery voltage
Log	Recorded file of a series of flight data to be examined later
Logical switch	Test on the data that can switch on functions
Lua	Programming language from Brazil used in OpenTx and elsewhere
NiMH	Nicket metal hydride. Battery used for Rx and Tx power.
mAh	Energy capacity of a battery in milliamp-hours. See also Ah.
Master	Trainer’s transmitter that controls the model. See also Slave
MicroSD	Small memory card in Tx with backups and other files
Mixer	Combining inputs to produce a combined effect, eg flaperon
Model setup	Set of screens to allow models to be set up on the transmitter
Momentary switch	Two-position switch that springs back when released
Motor	Electric power source for model
OpenTx	Free Open Source software used by FrSky and others
Output	Control signal to a servo, ESC, undercarriage or air brake
Radio setup	Set of screens to allow the transmitter to be set up for all models
Range check	Reduced transmitter output to allow a test of the sensitivity of the radio
Rate	How much of the servo throw is used as a percentage, aka weight
Rotary	An infinitely variable Tx control in the form of a knob

RP-SMA	Reverse polarity SMA connector – see SMA
RSSI	Received Signal Strength Indication. Telemetry from Rx about signal
Rx	Abbreviation for receiver
RxBt	Supply voltage at the receiver
S.BUS	Serial data bus to control all devices from a single receiver socket
Screenshot	Recording the Tx screen for later study
Servo	Device that moves the control surfaces
Slave	Trainee's transmitter that takes over control of model from the Master
Slider	An infinitely variable Tx control
SMA	Screw on coaxial radio aerial connector – Sub-Miniature version A
SmartPort	FrSky system for connecting telemetry sensors to receiver
Special function	An action that applies to one of the models stored on the transmitter
Subtrim	Moves the centre point of the servo throw, effectively permanent trim
tada	Voice message heard when the OpenTx transmitter is switched on
Taranis	FrSky transmitter that uses OpenTx software
Telemetry	Receiver sending data back to the transmitter, eg altitude and speed
Thermal	Up-current of warm air in which birds and gliders climb
Trainer	See buddy box
Trim	Small adjustments to control surfaces to make the model fly neutrally
Tx	Abbreviation for transmitter
UBEC	Universal battery elimination circuit – BEC on a separate device
Vario	See variometer
Variometer	aka vario Sensor for altitude & vertical speed using air pressure
Version	How up-to-date the software is, eg 2.2.2 (Major.minor.small change)
Weight	See Rate

List of useful websites

www.bmfa.org	British Model Flying Association
www.open-tx.org/downloads.html	OpenTx site for forums and documents
http://www.audacityteam.org/download/	Download audacity from here
http://www.rclife.co.uk/	Excellent supplier of FrSky kit
www.t9hobbysport.com/	Another excellent supplier of FrSky kit
www.peterscott.website	My website
peter@peterscott.website	My email address
http://open-txu.org/home/continuing-education/taranis-sd-card-addendum/	Excellent page on SD card sounds
www.barcs.co.uk	British Association of Radio Control Soarers
www.largemodelassociation.com	Large Model Association

Current hardware problems and weaknesses

There is only one problem that I know of, plus a weakness.

I think the Tx might be sensitive to a nearby mobile phone. On two occasions I forgot to leave my phone in my car. It was in my pocket and I suffered mysterious losses of control, probably during routine phone handshake signals with the base. I am happy to admit when it is pilot error and these weren't.

After over two years of intensive use I have only come across one design weakness. My **+** navigation button started to work intermittently then stopped altogether. I dismantled the transmitter and discovered that the three buttons are on a plastic strip that glues onto two pins. The underside of each button was a plastic spike that pushed a microswitch on the main circuit board. I reconnected the battery, switched on and, taking care not to short anything, I tested the relevant switch. It worked fine, so obviously the spike had worn. I had already bought a replacement button strip but removing the old one one looked tricky. So I put a dab of thick CA glue on the end of the spike with some accelerator. When it had hardened I pushed the board into place. The spike was now too long so I shaved slices off it with a scalpel until the switch worked again. I don't know how long it will last but it works.

Other points to note

Fragility

The transmitter and receivers are well built. To keep them light, some of the telemetry and S.BUS devices are fragile. I have taken to enclosing them in transparent battery heat shrink but you still need to be gentle with them.

Lack of co-ordination on documentation

Sometimes there are slip-ups with documentation. The most recent case was the manual for the Servo Channel Changer, which showed a totally different screen from the one on the delivered device. Delivered was Rev1 and the manual showed Rev1.1. It turns out that the device was never revised to Rev1.1 as this was for FrSky S.BUS servos that were cancelled. However the manual was still sent out.

Lack of UK repair facilities

UK dealers are only suppliers. They cannot carry out fault finding nor repairs. However spares, such as switches, gimbals and boards, are usually available and are cheap. If you can build a model you can do the repairs yourself by replacement. There will be someone in your club with test equipment if you haven't any.

Built-in sound tracks

These are in files stored on the SD card on the **SDCARD** page. To play one:
Move to the **SOUNDS** folder then **en**.
Select a file
Long press **ENT**
Press **Play**

Sound tracks in SOUNDS/en

These were from V1.6 but appear to be the same in V2.2

acro	acro mode on	fm-thml	flight mode thermal left
crowof	crow brake off	fm-thmr	flight mode thermal right
crowon	crow brake on	geardn	gear down
engoff	engine off	gearup	gear up
flapdn	flaps down	lnding	landing
flapup	flaps up	lowbat	low battery
fm -1 - fm -8	flight mode XX	nrmmod	normal mode on
fm-acr	flight mode acro	sigcrt	out of signal critical
fm-crs	flight mode cruise	siglow	out of signal low
fm-flt	flight mode float	spdmod	high speed mode active
fm-fst	flight mode fast	splrdn	spoiler down
fm-lch	flight mode launch	splrup	spoiler up
fm-lnd	flight mode land	thmmod	thermal mode on
fm-nrm	flight mode normal	tohigh	too high
fm-png	flight mode ping	tolow	too low
fm-pwr	flight mode power	trnoff	trainer off
fm-rce	flight mode race	trnon	trainer on
fm-spd	flight mode speed	vrioff	vario off
fm-thm	flight mode thermal	vrion	vario off

System sounds in SOUNDS/en/SYSTEM

Name.wav	**Message**		
		124	kmh
		125	metre
1 – 100	1 – 100	126	metres
101	200	127	degree
102	300	128	degrees
103	400	129	percent
104	500	130	percent
105	600	131	milliamp
106	700	132	milliamps
107	800	133	mAh
108	900	134	mAh
109	1000	135	watt
111	minus	136	watts
112	point	137	db
115	volt	138	db
116	volts	140	feet
117	amp	141	knot
118	amps	142	knots
119	metre per second	143	hour
120	metres per second	144	hours
123	kmh	145	minute

146	minutes	acro	acro mode on
147	second	crowof	crow brake off
148	seconds	crowon	crow brake on
149	rpm	engoff	engine off
150	rpm	flapdn	flaps down
151	g	flapup	flaps up
152	g	fm -1 - fm -8	flight mode XX
160	point zero	fm-acr	flight mode acro
161	point 1	fm-crs	flight mode cruise
162	point 2	fm-flt	flight mode float
163	point 3	fm-fst	flight mode fast
164	point 4	fm-lch	flight mode launch
165	point 5	fm-lnd	flight mode land
166	point 6	fm-nrm	flight mode normal
167	point 7	fm-png	flight mode ping
168	point 8	fm-pwr	flight mode power
169	point 9	fm-rce	flight mode race
(New 0170 - 0176)		fm-spd	flight mode speed
al_org	A1 Low	fm-thm	flight mode thermal
al_red	A1 Critical	fm-thml	flight mode thermal left
a2_org	A2 Low	fm-thmr	flight mode thermal right
a2_red	A2 Critical	geardn	gear down
endtrim	Maximum Trim Reached	gearup	gear up
inactiv	Inactivity Alarm	lnding	landing
lowbatt	Low Battery	lowbat	low battery
midtrim	Trim Center	nrmmod	normal mode on
rssi_org	RF Signal Low	sigcrt	out of signal critical
rssi_red	RF Signal Critical	siglow	out of signal low
swalert	Switch Warning	spdmod	high speed mode active
swr_red	Radio Antenna Defective	splrdn	spoiler down
tada	Welcome	splrup	spoiler up
telemko	Telemetry Lost	thmmod	thermal mode on
telemok	Telemetry Recovered	tohigh	too high
thralert	Throttle Warning	tolow	too low
timerl0	10 seconds	trnoff	trainer off
timer20	20 seconds	trnon	trainer on
timer30	30 seconds	vrioff	vario off
timerlt3	buzzer	vrion	vario off

As described above, you can add your own or change the standard ones. The how-to is in the open-tx link listed under ‘useful websites’.

This is the full set of over 450 sound **.wav** files in the alternative sounds pack from which you can choose. It is also a different female voice - sultry and clear. The sounds are read by someone called Amber, for which thanks. This list is for V1.6 and might be different in 2.2. I haven’t checked yet.

3dpact	3D pitch active	acrmd	acro mode
3dpitch	3D pitch	acro	acro mode on
3dpoff	3D pitch off	actv	active
3drtact	3D rates active	actvd	activated
3drtoff	3D rates off	aero	aerobatics
3drton	3D rates on	aero3d	3D aerobatics
3drud	3D rudder	aeropr	precision aerobatics
abrkcl	airbrakes closed	again	Oh, do that again
abrkop	airbrakes open	ail3d	3D ailerons

ailhgh	aileron high
aillow	aileron low
ailmed	aileron medium
airbk	airbrake
airbkoff	airbrake off
airspd	airspeed
althld	altitude hold
atmtc	automatic
atmtcmd	automatic mode
atti	altitude (should be attitude?)
attmd	attitude mode
auto	auto
autort	autorotation
bad	bad
badalt	bad altitude
badatt	bad attitude
batcrit	battery critical
batlow	low battery
battchrg	Hey. If its not too much to ask. Can you plug your charger into me? I’m in need of some good … electrons
battcns	battery consumption
bombawy	bombs away
bombrel	bombs released
brksoff	brakes off
brkson	brakes on
btflyoff	butterfly off
btflyon	butterfly on
camber	camber
camcntr	camber centred
camfix	camber fixed
camman	camber manual
clmb	climbing
clmbmd	climbing mode
clrprop	clear prop
colder	colder
corslk	course lock
corslkm	course lock mode
cpltoff	co-pilot off
cplton	co-pilot on
crcl	circle
crclmd	circle mode
crowoff	crow off
crowon	crow on
cruz	cruise
cruzmd	cruise mode
currdrw	current draw
dactvd	deactivated
deplyd	deployed
dlg	d l g
dlgf	discus-launch glider
dscnd	descending
dseng	disengage
ductf	ducted fan
ele3d	3D elevator
elehgh	elevator high
elelow	elevator low
elemed	elevator medium
eng	engage
engarm	engines armed
engdis	engine disabled
engdisa	engines disarmed
engoff	engine off
engon	engine on
eww	ew nasty
extmodl	external module
fandf	fast and flat
fbwa	fly-by-wire a
fbwb	fly-by-wire b
flapfll	flaps full
flaphlf	flaps half
flaps1	flaps 1
flaps2	flaps 2
flapsdep	flaps full
flapsdn	flaps down
flapsup	flaps up
flkswtch	if you flick that switch one more time ...
flkswtch2	mmm. I’ll let you flick that switch again
flpail	flaps synced with ailerons
flpnorm	flaps normal
fltmode1	flight mode 1
fltmode2	flight mode 2
fltmode3	flight mode 3
fltmode4	flight mode 4
fltmode5	flight mode 5
fltmode6	flight mode 6
fltmode7	flight mode 7
fltmode8	flight mode 8
flypst	you wanna fly that past me one more time?
fpv	f p v
fpvf	from pilot’s view
frget	aren’t you forgetting something?
Frget2	I think you’re forgetting something
fsoff	failsafe off
fson	failsafe on
fxdwng	fixed wing
geardn	gear down
gearup	gear up
getreal	things are about to get real
ggl	(sound of a giggle)
gglno	(sound of a giggle) then no
gglyes	(sound of a giggle) then yeah
gmblact	gimbal active
gmbloff	gimbal off
gmblon	gimbal on
good	good
gps	g p s
gpsmd	gps mode
gyrhh	gyro heading hold
gyrrate	gyro rate mode
hedhld	heading hold active
heli	helicopter
heli3d	3D helicopter
heliscl	scale helicopter
hexc	hexacopter
hirates	high rates

homlk	home lock
homlkm	home lock mode
hotter	hotter
htoff	head track off
hton	head track on
hvr	hover
hvrmd	hover mode
idldwn	idle down
idlup	idle up
idlup1	idle up 1
idlup2	idle up 2
ignact	ignition active
ignoff	ignition off
ignon	ignition on
intmodl	internal module
ioc	i o c
iocoff	ioc off
jett	jet turbine
landing	landing
lhtsoff	lights off
lhtson	lights on
liketht	Oh I like that. Do it again
lnch	launch
lnchmd	launch mode
lndgmd	landing mode
loitr	loiter
loitrmd	loiter mode
lowrates	low rates
lwstcell	lowest cell
man	manual
manmd	manual mode
manual	manual
mdact	mode active
mdoff	mode off
mdon	mode on
meantto	I meant to do that
midrates	middle rates
mltrtr	multi-rotor
mode	mode
mode0	mode zero
mode1	mode 1
mode2	mode 2
mode3	mode 3
mode4	mode 4
mode5	mode 5
mode6	mode 6
mode7	mode 7
mode8	mode 8
navltoff	navigation lights off
navlton	navigation lights on
neg	negative
ngtvghst	negative ghost writer the pattern is full ????
normal	normal
notsuk	that didn't suck at all
nottht1	not that one
nrmmod	normal mode on
nxtmlwr	next time a little lower
nxttmhr	next time a little higher
octc	octocopter
off	off
ohnodnt	oh no you didn't
on	on
oops	oops
opentx	open tx
packvlt	pack voltage
pchtdep	parachute deployed
pos	positive
poshld	position hold
pwrgld	power glider
pwrrd	power reading
pylnrc	pylon racing
quadc	quad copter
race	race
racemd	race mode
racing	racing
ratemd	rate mode active
reflex	reflex
rtl	return to launch
rudhgh	rudder high
rudlow	rudder low
rudmed	rudder medium
sclmod	scale model
sclpitch	scale pitch
seapl	seaplane
seek	seek
seekmd	seek mode
sgnlcrt	signal critical
sgnllow	signal low
sim	simulator
sitnorm	situation normal
situ	situation
slflvl	self-level mode
slflvlm	self-level mixing mode
sltscl	slats closed
sltsop	slats open
smokeoff	smoke off
smokeon	smoke on
snapsw	snap roll
soring	soaring
speed	speed
speedmd	high speed mode active ???
splrcl	spoiler closed
splrop	spoiler open
sprtsfl	sports flying
srf_ail	aileron
srf_ails	ailerons
srf_elev	elevator
srf_rud	rudder
srf_thr	throttle
stbloff	stabilisation off
stblon	stabilisation on
stblz	stabilise
stblzmd	stabilise mode
stntact	stunt mode active
stntoff	stunt mode off
stnton	stunt mode on
syson	system on ready to fly
takoff	take off
thnks	why thank you

thract	throttle active	turnon	mmm you turn me on
thrdis	throttle disabled	uav	u a v
thrhold	throttle hold	ugly	ugly
thrml	thermal	usrerr	user error. Please replace user and try again
thrmlmd	thermal mode	vintg	vintage
thrrel	throttle release	vriooff	vario off
thtsw	oh yeah that was the right switch	vrioon	vario on
tlrtract	tail rotor active	warbrd	warbird
tlrtroff	tail rotor off	wasmnt	wasn’t meant to do that
toohgh	too high	waypt	waypoint
toolow	too low	whtvr	whatever
train	training	yrwelcm	you’re welcome
tric	tricopter	zoom	zoom
trninst	instructors plane	zoommd	zoom mode
trnr	trainer		
trnstd	students plane		

An example of a fairly complicated setup: glider ‘full-house’

‘Full-house’ means that the flaps and ailerons work differently for different flight modes. For normal flying the flaps are neutral and the ailerons work as usual with more up than down movement. For thermal soaring, flaps and ailerons are moved down by up to 10º using a rotary control to add undercamber. The ailerons become flaperons. For high speed aerobatics, flaps and ailerons are moved up a little to make the airfoil a bit more ‘slippery’. For landing on tight slopes the flaps go up and flaperons go down to create crow brakes. This is relatively easy to do on the Taranis as mixes are very flexible. There are four rotary controls and the code lines allow you to do almost anything. The battery voltage is read out when not at zero throttle and a stronger warning is given at about 30% capacity.

Channels
This is how the channels are used:
1 Throttle with the ESC set to brake on zero setting to allow the propellors to fold.
2 Left aileron with up movement double that of down.
3 Elevator
4 Rudder
5 Right aileron with up movement double that of down.
6 Flaps using a Y-lead.

Telemetry
Connected sensor is a 40A current sensor providing **Curr** and **VFAS**
New sensor **mAh** must be created as described above.

Controls
On the Taranis X9D plus transmitter I use the following switches and rotaries:
Three position switch **SD**: forward - vario tones, centre - sound off, back - height readout
Three position switch **SC**: forward motor on, other positions motor off, with voice message
Rotary **RS**: voice and vario volume
Rotary **S2**: crow brake setting
Rotary **S1**: undercamber or slippery flap/flaperon settings
S1 and S2 will both be moved using my left hand on mode 2.
Two position switch **SF**: forward – normal rates, back – high rates
Three position switch **SE**: centre – reset vario data to zero field altitude

These are the lines of code used in the transmitter. You can use different numbers for the functions (GF, L and SF).

Functions

Global functions

GF1	ON	Volume	RS	☑	(allows the voice volume to be changed)

Logical switches

L01	a>x	Alt	122.0m	(switches on when Alt more than 122m)
L02	a>x	□Thr	-90	(tells pilot battery voltage when throttle not -100)
L03	a>x	mAh	950mAh	(Switches when battery getting low; 1300mAh battery)

Special functions

SF1	SD↑	Vario			
SF2	SD↓	PlayValue	Alt	10s	(says altitude every ten seconds)

SF3	L01	PlayValue	Alt	10s	(looks at logical switch L01)
SF4	SE-	Reset		Telemetry	▯
SF5	SE-	Play Sound		ratata	1x
SF6	SC-	OverrideCH1		-100	▯
SF7	SC↓	OverrideCH1		-100	▯
SF8	SC-	Play Track		engoff	1x
SF9	SC↓	Play Track		engoff	1x
SF10	L02	Play Value		VFAS	20s (set it to one decimal place)
SF11	L03	Play Value	mAh	10s	(Speaks mAh value every 10s after 1500)

Inputs, mixes, outputs and telemetry

Note: the numbers in the following lines are not fixed. They were the ones I needed for my servo/control surface horn and linkage geometry. You will need different values.

Inputs

Switch SF gives the dual rates

I Thr	100	▯	Thr		---
I Ail	50	▯	Ail	E30	SF↑
	100	▯	Ail		SF↓
I Ele	85	▯	Ele	E30	SF↑
	100	▯	Ele		SF↓
I Rud	100	▯	Rud		---
I Flap	100	▯	S1		---
I Crow	100	▯	S2		---

Mixer

CH1	100	I	Thr
CH2	100	I	Ail
+=	20	I	Flap
+=	-80	I	Crow
CH3	100	I	Ele
CH4	100	I	Rud
CH5	100	I	Ail
+=	20	I	Flap
+=	-80	I	Crow
CH6	20	I	Flap
+=	-80	I	Crow

Outputs

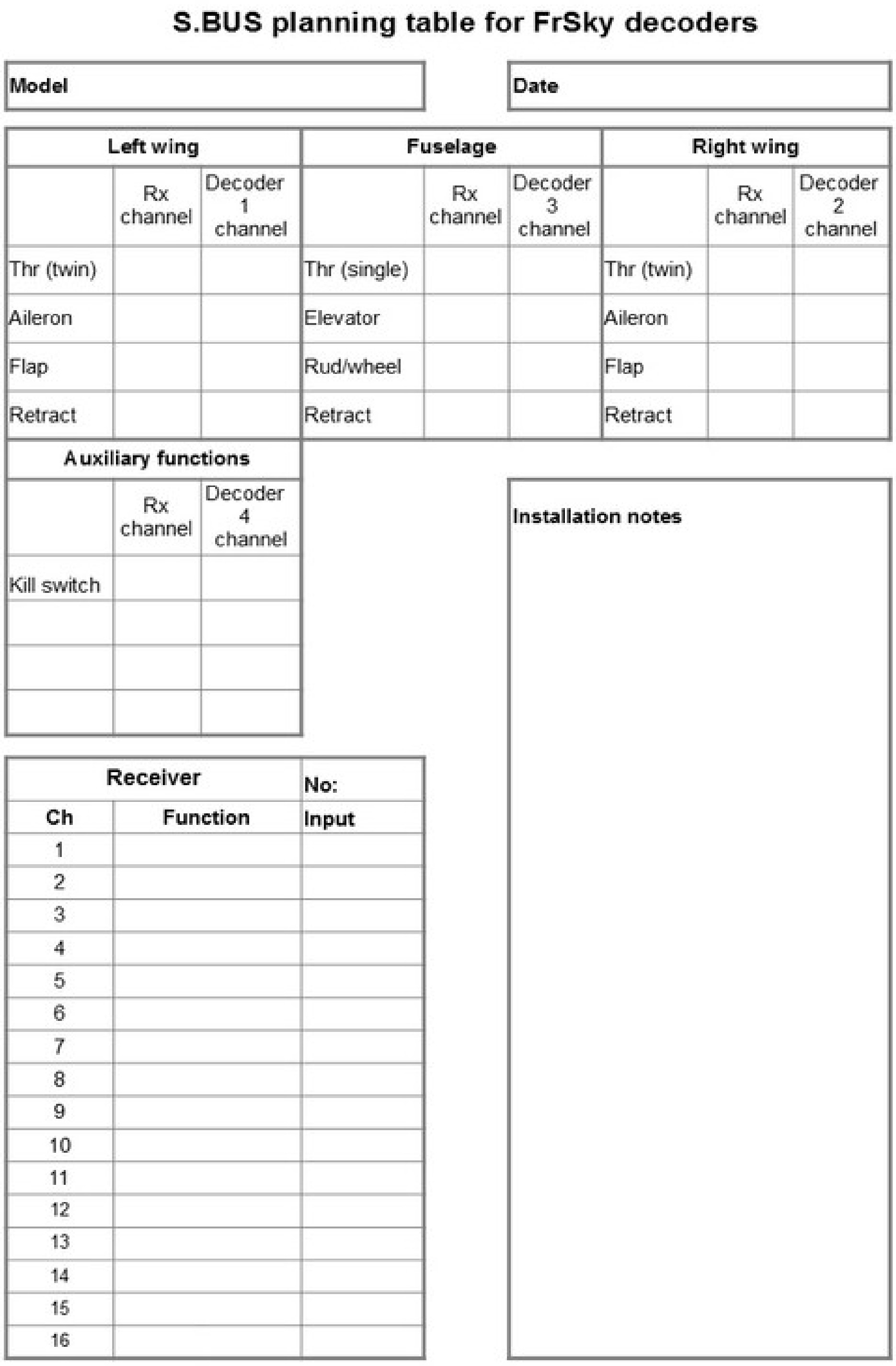
These lines give the aileron differentials and reversal

CH2	0.0	▬	-70	▬	35	▯	---	1500	▲
CH5	0.0	▬	-70	▬	35	▯	---	1500	▲

Model setup

Model image	Fox glider		Best match for my Rider soarer
Timer 1	Ths	03:00	Sets timer to 3 minutes of throttle open
Countdown	Voice		Speaks minutes, then seconds in last minute

S.BUS decoder planning tables



S.BUS planning table for FrSky decoders

Model	**Date**

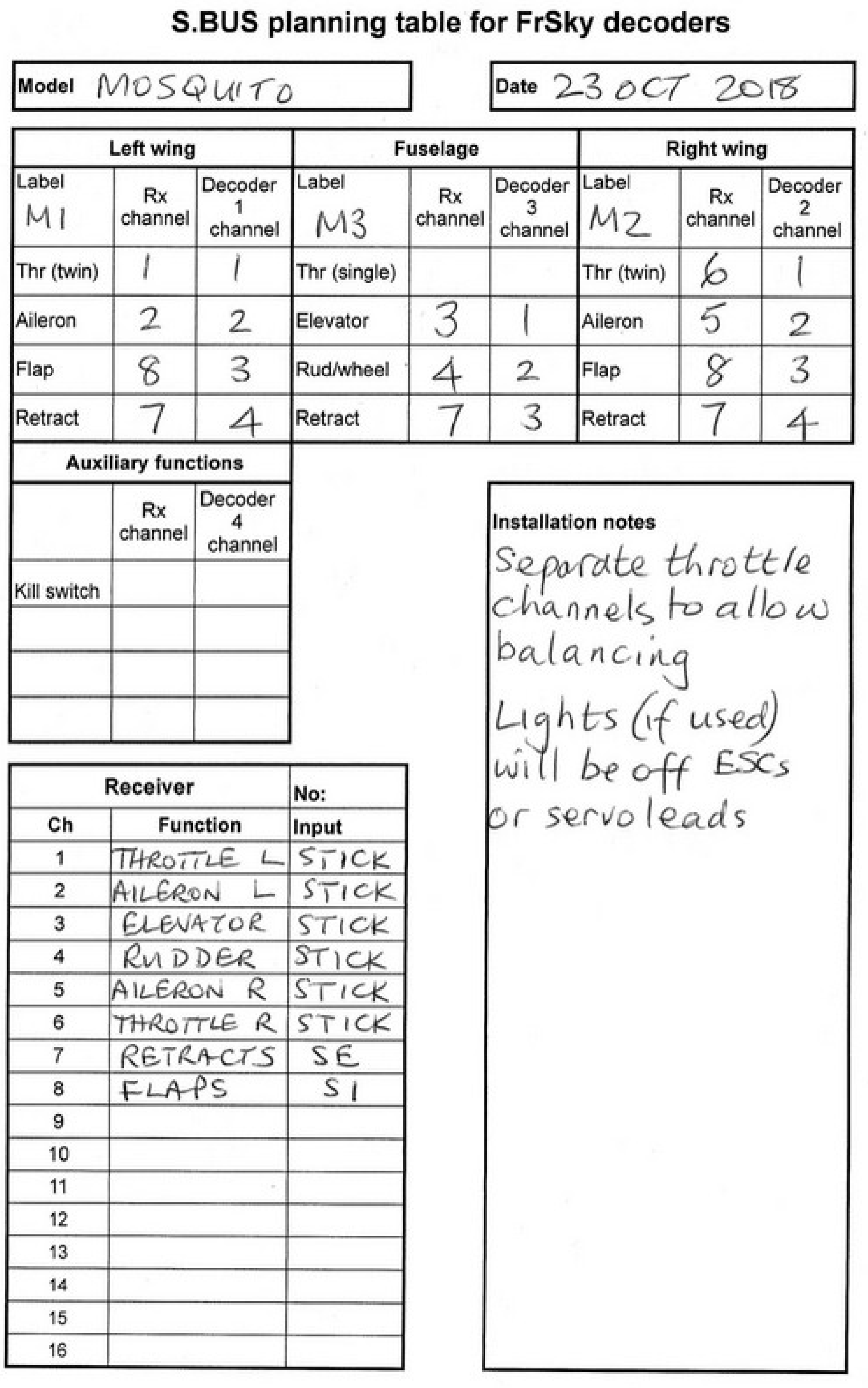
Left wing			**Fuselage**			**Right wing**		
	Rx channel	Decoder 1 channel		Rx channel	Decoder 3 channel		Rx channel	Decoder 2 channel
Thr (twin)			Thr (single)			Thr (twin)		
Aileron			Elevator			Aileron		
Flap			Rud/wheel			Flap		
Retract			Retract			Retract		

Auxiliary functions		
	Rx channel	Decoder 4 channel
Kill switch		

Receiver		**No:**
Ch	**Function**	**Input**
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		

Installation notes

This is the planner used for my Mosquito model



S.BUS planning table for FrSky decoders

Model MOSQUITO	Date 23 OCT 2018

Left wing			Fuselage			Right wing		
Label M1	Rx channel	Decoder 1 channel	Label M3	Rx channel	Decoder 3 channel	Label M2	Rx channel	Decoder 2 channel
Thr (twin)	1	1	Thr (single)			Thr (twin)	6	1
Aileron	2	2	Elevator	3	1	Aileron	5	2
Flap	8	3	Rud/wheel	4	2	Flap	8	3
Retract	7	4	Retract	7	3	Retract	7	4

Auxiliary functions		
	Rx channel	Decoder 4 channel
Kill switch		

Receiver		No:
Ch	Function	Input
1	THROTTLE L	STICK
2	AILERON L	STICK
3	ELEVATOR	STICK
4	RUDDER	STICK
5	AILERON R	STICK
6	THROTTLE R	STICK
7	RETRACTS	SE
8	FLAPS	SI
9		
10		
11		
12		
13		
14		
15		
16		

Installation notes

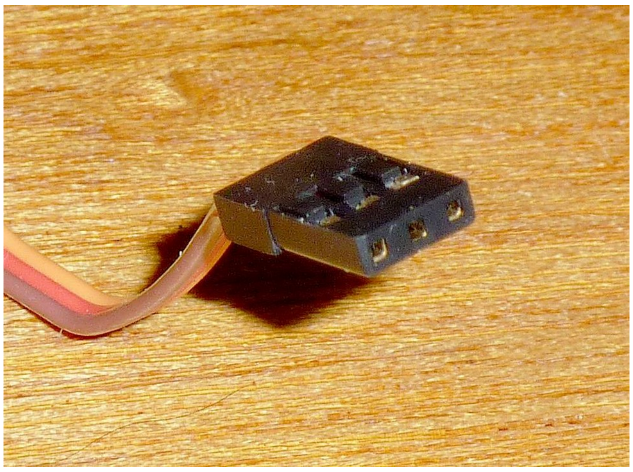
Separate throttle channels to allow balancing

Lights (if used) will be off ESCs or servo leads

OpenTx telemetry (version 2.1 on)

The current telemetry system is the result of a complete rewrite in 2015. The more I learn about it the more impressed I am. As described above, setting up telemetry for such as lipo voltage, variometer, airspeed indicator and GPS sensors is simple. However stopping there is missing the great strength and flexibility of the system. To help with the hardware costs of the FrSky project I send a modest annual donation.

The telemetry sensors run on a bus system similar to S.BUS. They connect in a daisy chain, connecting one to the next and so on, using female to female servo-type leads that finish in a single SmartPort port on the receiver (Rx). Note that I use ‘female’ here in the correct electronics meaning of metal sockets not metal pins in the connectors.


Female servo connector

Real and virtual sensors

Confusingly the same name ‘sensor’ is used for both the real physical devices and the data they produce.

Real/physical sensors (electronics)

These are the actual electronic devices. Each has an ID. The default values can be changed using an S.BUS servo channel changer. You might never need to do this, but if for example you want to monitor the rpm and temperature of two motors and ESCs the second sensor must have a different ID. There are actually two IDs for each real sensor. The first is an ID for the device and has four digits, e.g. F210. This can’t be changed so can be duplicated. The second is the ID used by the Rx to identify the sensor, 1 to 32. This must be unique for each model setup.

Physical sensors produce one or more data. The system calls each datum a ‘virtual sensor’.

Virtual sensors (data)

There are actually two types:

- Data produced by the real sensors
- Data calculated from one or more other data

For clarity I will use the words ‘datum/data’ rather than the term ‘virtual sensor(s)’. Each datum can be reset individually using a special function and a real or logical switch, or you reset all at the same time using a special function.

1 Data produced by the real sensors – ‘custom’ data

Each datum has a name, unit and value. These are some of the ‘properties’ of the datum that can be changed using the **Edit** function (more below). One datum can be displayed more than once, for example to display height in both metres and feet.

2 ‘Calculated’ data

Amongst other things, data can be added, averaged or multiplied, and the minimum or maximum of up to four data can be found. To do this you click **Add a new sensor**. This opens an edit screen. You give the data a name then set **Type** to **Calculated**. For example the motor’s voltage multiplied by the current gives the power. Don’t forget the calculated data are still called ‘virtual sensors’. Editing is described later.

Using data

You can see data on one of the telemetry screens, hear it read out to you at chosen intervals or use it as a logical switch to trigger a function. You could even use it as an input, for example to change elevator trim settings depending on airspeed or opening an air intake grill to cool an overheating motor or ESC.

Editing a real or virtual sensor

Move to the screen that lists the telemetry sensors.
Select the sensor to edit or create a new one.
Long press on **ENT** and select **Edit**.
Type: Choose **Custom** or **Calculated** and click **ENT**

This is the list of things you can edit:

If **Custom** is selected:

Name:	Change it if more than one of the same sensor.
Type:	Reads **Custom**
ID:	First is fixed. Don’t change the second if already sensed unless a duplicate.
Unit:	Use this to select, for example, m or ft for height.
Precision:	Sets the number of decimal places. Useful for when value is read out.
Ratio:	Multiplier to use to get the correct reading, e.g. for a voltage checked by voltmeter.
Offset:	Sets initial value to non-zero, e.g. for GAlt when airfield not at sea level.
Positive:	Ignores negative values, e.g. no vario sound when sinking.
Persist.:	This tells the transmitter to retain the value from previous sessions, e.g. Aspd+ maximum ever airspeed for that model.
Logs:	This tells the transmitter to store the data stream on your memory card.

If **Calculated** is selected some are different:

Type:	Reads **Calculated**
Formula:	Choose from **Add, Average, Min, Max, Multiply, Totalize, Cell, Consumpt, Distance.** (See more details below)
Source1 to 4:	The lists will contain all of the data items currently in telemetry
Filter:	By averaging, it smooths values that change a lot , e.g. lipo voltage
Auto Offset:	This sets the first received value as the zero when reset or at switch-on.

Formulae explained

Add, Average, Min, Max and Multiply are obvious.

Totalize: This adds the sensor value to running total

Cell: The value of an individual cell is found using:

Cell Index:	Number showing position in cell series
Lowest:	Obvious
Highest:	Obvious
Delta:	Highest minus lowest voltage

Consumpt: mAh used using current sensor data **Curr**

Distance: Distance between the pilot and the model using GPS

Telemetry sensors with default IDs and data items

Sensor	ID	Data items
Air speed indicators (2))	10	Aspd
Current sensors (2)	3	VPAS (40A only), Curr
Liquid fuel monitoring	Not yet known	
GPS	4	GPS, Galt, Gspd, Hdg, Date, Time
Lipo voltage sensors (3)	2	Cels
RPM and temperatures	5	RPM, Tmp1, Tmp2 (see note below)
Receiver	25	RSSI, RxBt, SWR
Variometers (2))	1	Alt, Vsp
Neuron ESCs	17	EscV (volts), EscA (current), EscR (RPM), EscC (consumpt), EscT (ESC temperature)

Plus new sensor hub with sockets for even more compact sensors for: ID not yet known

- Fuel Gauge
- GPS
- Variometer
- Lipo Voltage
- Temperature
- Power Supply
- RPM Sensor
- Triaxial Accelerometer

By the way, when setting up an RPM sensor either set the number of coils on the sensor device card or in the sensor edit screen, but not both.

SP2UARTs

SP2UART (universal asynchronous receiver/transmitter) devices are like modems. You need a pair, one in the Rx and the other connected to another the read/write device. They can transmit and receive analogue and RS232 serial data at up to 9600 baud (that takes you back a bit doesn't it?) but usually at no more than 300 baud. Yes 300 bit/s. I could imagine it being used in robotics but I can't yet see a use in model aircraft. When I come across a practical application that I understand enough that I can describe it, I'll add it here The data they produce are called A3 to A6.

A3/A4 SP2UART host	ID = 6
A5/A6 SP2UART remote	ID = 7

FrSky ('free-sky') Neuron ESC

FrSky has released a new range of electronic speed controllers for 3S to 6S batteries. Called Neuron they come in 40, 60 and 80 capacities with 50% more peak current. They have a voltage adjustable 7A SBEC. The reason they are mentioned here is that they have built-in telemetry sensors for voltage, current, RPM, power consumption and temperature which plug into the receiver SmartPort. There is much more about these devices below.

First impressions were very good. The ESC comes packed in foam in a solid plastic box. It is very strongly made with a thick aluminium plate top and bottom. All sides are open. There are two servo-style sockets to connect to the receiver – one for the throttle/BEC lead (PWM) and the other for the Smartport (S.Port). You need leads with both ends female (socket rather than pin). You will need to cut the red core on the PWM lead if using a separate receiver battery, but this is a separate lead so you are not cutting one that is permanently connected to the ESC. The ESCs are compact with good heat sinks and are available in 40, 60 and 80 amp versions with 50% extra capacity for short times (burst). All versions are the same size and weight, though different prices, and can be connected to 3S to 6S batteries.

After soldering on XT90 and 4mm bullet connectors and sleeving, the device weighed 73g. This is exactly the same as a Turnigy Plush 60A, though of course the latter has no telemetry. The sizes are: Neuron: 60 x 33 x 16 mm Plush: 72 x 30 x 17 mm so the Neuron is just a bit shorter. Other ESCs are available.

Clearly this is designed with racing quads in mind, as it is marked BLheli. More about BLHeli later. However I will use it on fixed wing.

The device includes a range of FrSky telemetry. I tested the telemetry using a Taranis X9D plus transmitter running OpenTx V2.2.2 and a freestanding X8R receiver. Initially no motor was connected, so the current, rpm and mAh consumption data were zero. I allowed the ESC to power up the receiver through the BEC and the voltage shown in RxBt was 4.9V. Using a voltmeter I checked whether this was the voltage sent by the BEC and it was, so the BEC appears to default to 5V. It can provide 7A. The voltage can be changed. Much more of that later.

After using ‘Discover new sensors’ on the Taranis all of the data appeared as follows:

Names	Values during test.		
0E50	2560804	This stores encrypted BEC values whatever that means.	
EscV	16.71V	Lipo voltage	Volt
EscA	0.00A	Motor current	Amp
EscR	0rpm	Motor speed	RPM
EscC	0mAh	Consumpt – mAh used	mAh
EscT	44°C	ESC temperature	Temp

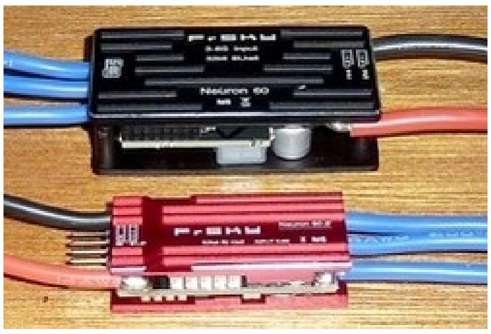
As you see the data (sensors) have different names from the ones created by separate telemetry devices. They seem sensible but I changed them to those shown in the last column above. This makes the telemetry screen more understandable. All have the same device ID (17). If using two of these in a twin motor model you will have to change the device ID for one set of devices.

I then added the new data to a numeric telemetry screen which worked perfectly. I decided also to add Amp+ and RPM+ so that I can have a reading of the maximum current and motor speed during the flight. The former will be essential to make sure that I have the correct propellor fitted. To get maximum safe power I want about 90% of the maximum current for the motor when the prop is unloaded in the air.

Then I connected a sizable 4Max motor and ran it up, hand held, without a propellor. Sensors Amp, RPM, mAh now generated data. The rpm one showed over 30000 rpm which puzzled me until I remembered that rpm has to be calibrated for the number of coils in the motor, and defaults to one. I’ve edited the sensor to the 12 coils for the Eflite Power 46 that it will be connected to when I install the ESC in a model. The defaults appear to work fine for fixed wing, though I think braking is set on as the motor stopped sharply.

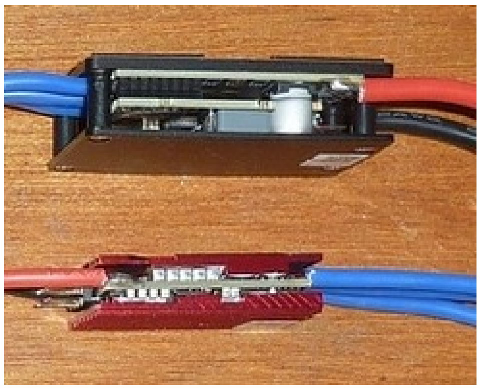
Neuron S versions

In late 2019 FrSky issued updated versions of the Neurons, designated ‘S’ (small?). I bought a 60S and a 40S. They are very much smaller as you can see from the photographs comparing the 60 and the 60S. FrSky has done an amazing job squeezing the speed control circuitry and the telemetry into such a small device. The S has a jumper to select whether the BEC is used so you don’t need to cut the red core on the throttle lead if it isn’t. One last bonus was that the throttle did not need calibrating. It presumably defaults to 1000 to 2000 ms.



60	76g	60 x 33 x 16 mm
60S	47g	45 x 22 x 12 mm

In both cases weights are with one XT90 and three bullet connectors



I have installed the 40S in my Acrowot foam-e. It fitted perfectly in the original position. I don’t need to connect the battery balance lead to a voltage sensor for voltage telemetry so the wiring is much neater. And now I know when I’ve used 1500 of the 2200 mAh in the battery cos the nice lady tells me. I used the default 5V BEC on the 40S.

I fitted the 60S in my Wot4. It was much smaller and lighter than the Turnigy Plush ESC so I was able to put it in a more convenient place and add a NiMH receiver battery and switch. As always, in both cases the telemetry data was found by the Taranis without a problem.

What’s more, for now you get a free USB linker toolstick with the S for use with the BLHeliSuite software to program the ESC if you need to, which you probably don’t for fixed wing.

A word of warning

I have found that the S versions of Neurons are more fragile electronically. If you misconnect one it is likely to burst into flames so check very carefully before powering up. Cooling seems to be very critical too. I had to replace the 40S I installed in my Acrowot as under full power it started to cause the motor to judder. Unfortunately I hadn't set the telemetry to retain the maximum temperature so I don't know if that was the problem

Advantages

There are three: size, weight and cost

Size: The dimensions of the Neuron are good compared with other makes, and no further sensors are needed so the whole setup takes up much less space. Glider pilots might want the separate vario and GPS sensors, though the former is now built in to some FrSky receivers.

Weight: The Neuron is no heavier than most ESCs and you will not need to install other sensors with their associated wires. To match the Neuron you need lipo voltage, current and speed/temperature sensors which add up to about 27g plus wires. We are not told what the current sensor will read up to, but I assume that it will be at least the current capacity of the ESC.

Cost (Sept 19 prices): Neuron 40 £45.60, **Neuron 60** £55.20, **Neuron 80** £63.60. Unless you need an air speed, vario or GPS sensor, that's it. The cheapest FrSky sensors are: lipo £10.44 (2.8g), current £17.00 (17g), rpm/temp £14.50 (6.7g). That's another £42.

What don't I like?

It's not often I get cross with FrSky but this time I am. Much of the manual is concerned with setting up quads, so I guess that is the target use. However I want to use it in fixed wing models so all I need to set is the battery elimination circuit (BEC) voltage, and possibly motor timing and braking. The BEC default is 5V but you can set it to up to 8.4 for higher voltage servos. So what makes me cross? FrSky doesn't answer the question, 'how do you do it?'.

What does the ESC manual say? "The SBEC voltage can be adjusted through LUA (FrOS & OpenTX Supported) or through FreeLink App with Airlink S." Er, what? That's it. No instructions on how to do this. The ESCs are very new so there is little on the fora. There are many other parameters to set up but I am hoping that the defaults will work for fixed wing.

So using some logic on the manual information it appears that there are two suggested methods. The first is Lua, which is a programming language. I think there are simpler options, using computer software and USB devices. However I will pursue this later.

So what is the second option, FreeLink App and Airlink S? You need to buy a FrSky device called an S.Port Air Link-S device and read the manual. You also need to unzip and install the FreeLink app on your computer which can be downloaded from the FrSky site (https://www.frsky-rc.com/s-port-airlink/). The file is called 'frsky_update_Sport_rev14.zip'. Well I tried. And tried. But I could not get the connection to the ESC to work, so I put that to one side for now.

Third way: Using a BLHeli USB toolstick with Neuron ESCs

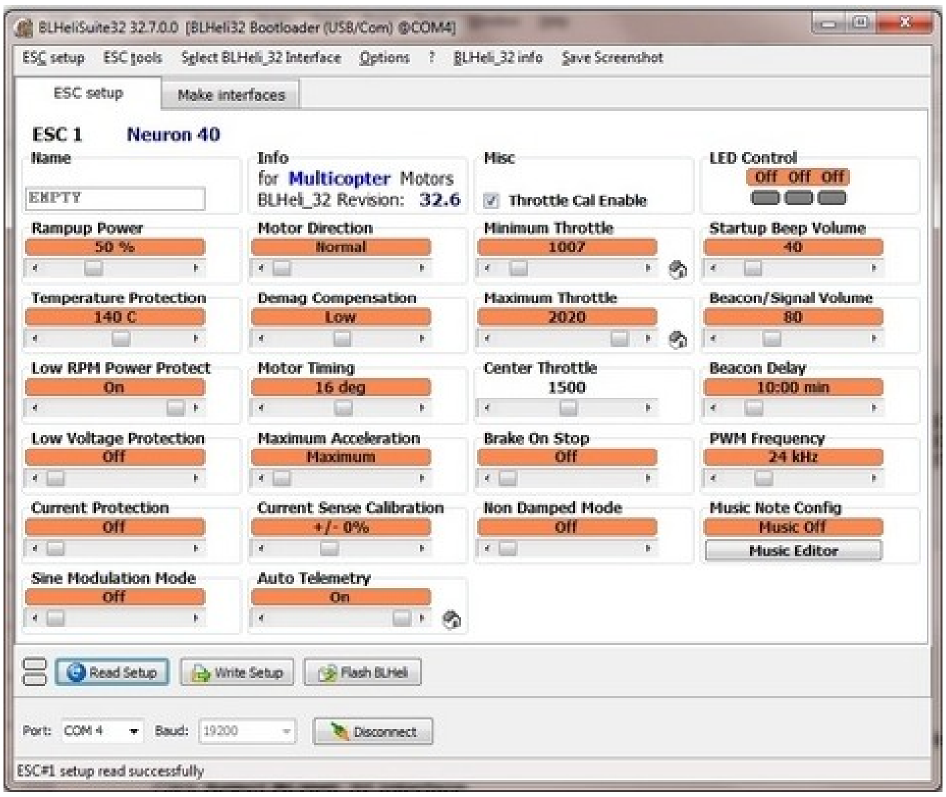
However there is a third way to set up the ESC. Richard Bago of T9hobbysport suggested using the BLHeli software to set the parameters. You have to buy a USB dongle called a **toolstick** for your computer that connects to the PWM connector on the ESC.

These toolsticks are sold under a variety of names from many sources.

You must install the free BLHeliSuite32 (9.6Mb). To download for Windows go to
http://www.mediafire.com/file/9ylhp0s0ces4tw2/BLHeliSuite32_32700.zip/file
(The site also has versions for other computers and devices.)

After getting your virus checker to scan it, unzip it.
Open it.
Plug in the toolstick.
Connect the toolstick to the PWM connector on the ESC using a female/female servo lead **with the red wire cut**.
Click **Select BLHeli_32 Interface**
Click on **BLHeli Bootloader (USB/Com)**
Select the Port, in this case COM 4. (This will be different for linux and Apple computers.)
Power up the ESC (connect a battery).
Click **Read Setup**
If all is well you get the message 'ESC#1 setup read successfully'
Click **OK**

You should now see the following screen. Notice the word **ESC1** and the name **Neuron** (confusingly Neuron 40 as mine was a 60)



BLHeliSuite32 32.7.0.0 [BLHeli32 Bootloader (USB/Com) @COM4]

ESC setup | ESC tools | Select BLHeli_32 Interface | Options | ? | BLHeli_32 info | Save Screenshot

ESC setup | Make interfaces

ESC 1 Neuron 40

Name: EMPTY

Info: for Multicopter Motors, BLHeli_32 Revision: 32.6

Misc: ☑ Throttle Cal Enable

LED Control: Off Off Off

Rampup Power: 50 %

Motor Direction: Normal

Minimum Throttle: 1007

Startup Beep Volume: 40

Temperature Protection: 140 C

Demag Compensation: Low

Maximum Throttle: 2020

Beacon/Signal Volume: 80

Low RPM Power Protect: On

Motor Timing: 16 deg

Center Throttle: 1500

Beacon Delay: 10:00 min

Low Voltage Protection: Off

Maximum Acceleration: Maximum

Brake On Stop: Off

PWM Frequency: 24 kHz

Current Protection: Off

Current Sense Calibration: +/- 0%

Non Damped Mode: Off

Music Note Config: Music Off

Music Editor

Sine Modulation Mode: Off

Auto Telemetry: On

Read Setup | Write Setup | Flash BLHeli

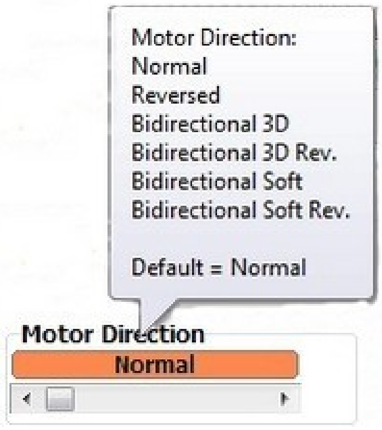
Port: COM 4 Baud: 19200 Disconnect

ESC#1 setup read successfully

Notice also the following:
I had already calibrated my transmitter throttle stick (see later) so the range is Minimum 1007 and the Maximum 2020 (milliseconds).

Brake On Stop is **Off.** For a folding prop on a glider you set this on.

Motor direction is **Normal**. If you click on the Normal button you get the following options, including centre-off for Bi-directional, possibly useful for float-planes to reverse out of reeds.



Motor Direction:
Normal
Reversed
Bidirectional 3D
Bidirectional 3D Rev.
Bidirectional Soft
Bidirectional Soft Rev.

Default = Normal

Motor Direction: Normal

At this point, as they wisely say in Norfolk, ‘Hold you hard’. You can change any of the settings then send them to the ESC by clicking **Write Setup**. But before you start, read the manual then read it again. To get the full manual click **BLHeli_32 info**.

For fixed wing you probably won’t need to change any of the more esoteric options. However possible candidates you might consider are:
Rampup Power
Low Voltage Protection
Current Protection
Motor Timing
Beacon

Another good idea is to use the **Save Screenshot** menu option. It will be a .png file similar to **BLHeliSuite32ESC setup_190925_1.png,** with a different date (YYMMDD) and version (190925_1). Then copy or print it. Then when you mess up you can easily reset the ESC to the original settings. Anyway it will be a good idea to save a screenshot when you are happy with the setup.

The fourth way: Throttle calibration using the transmitter

You might not need to use any of the above methods. Read on.

I fitted the ESC into my motor test bed, running a Turnigy 3542 motor on a 4S lipo battery. To start I used a self-powered servo tester to provide the throttle signal. I connected the battery. The ESC made quite a few beeps but when I pushed the tester to full throttle the motor turned quite slowly. Clearly throttle calibration is needed for this ESC.

The instructions came from the end of the BLHeli32 manual. Notice - not from the Neuron manual.

Push the throttle stick to maximum.	
Connect the battery.	
You then get – wait for it ...	
Three fast rising beeps	
One long low beep	(signal detected)
Four slow high beeps	(measuring setting)
Three sets of four fast rising beeps	(max throttle stored)
Silence	
Pull the throttle to minimum.	
Four slow lots of two low beeps (seemed more)	(measuring setting)
Three sets of four fast falling beeps	(min throttle stored)
Then the startup, arming tones:	
Three rising tones	(Power on)
One long low tone	(Signal detected)
One long high tone	(Zero throttle detected)

That’s almost enough to orchestrate as the theme of a symphony. I wonder what would happen if I fed it to the phone app that recognises tunes?

And then, on throttle up, the motor ran full chat. There was no need to disconnect the battery to reset it. Next I connected the Neuron to an X8R receiver on the test bed, powered with a separate battery. I had to go through the calibration again for the X9D transmitter throttle. All the telemetry sensors produced good data after discovery.

I am getting to like the tune. Maybe I'll write a rap to it celebrating the joys of FrSky. Then a Spektrum, Futaba or Hitec lover can write another and we'll do a battle rap on the flying field.

Oh yes. One tiny extra. According to the BLHeli manual you can set the throttle so it is centre zero and the motor can then go both forward and reverse. Why would you want to? If you fly off water you might want to reverse out of reeds, as you can with the reverse on the Tundra model. I don't yet know if you can do that with the Neurons though.

In summary this is a very impressive device that I will definitely be using for up to 6S setups. However, come on FrSky! You have created an excellent product but spoiled it with a useless manual. If these ESCs are to catch on you must make their setup much easier, ideally with a programming card like other ESCs. I got my Neuron from T9HobbySport as their service is excellent and delivery is normally next day.

Neuron 60 oddities

Some Neuron 60Ss will not give full power even if calibrated with the transmitter or through BLHeli Suite. The ESC it seems is over-protective. Go back into BLHeli Suite and set the following:

Set 'Low RPM Power Protect' to OFF
Set 'Ramp-up Power' to 150%

All should then be well. Don't forget to calibrate EscR to the correct number of motor coils.

Test flights

Setup
I fitted it in a Wot trainer with a separate receiver battery. This has an Eflite Power 46 turning a 13 x 8 prop. Current ratings are 40A continuous and 55A burst. I set up my telemetry screen to display current, maximum current, rpm, maximum rpm, consumpt (mAh used) and battery voltage. The battery was a fully charged 5 Ah 4S nanotech with internal resistances of about 3mΩ.

Current and power
Full throttle current was 55A static and 40 to 50A in the air. This rose to 52A in manoeuvres. This makes the maximum average power about 52 x 14.4 = 748 W. The spec shows 800W.
Cruise current was 20 to 30A in level flight and taxiing was around 15A.

Consumption
I also checked the consumption figures. I landed after nine minutes having used 3000mAh.

A meter showed 49% remaining, which was a bit high. The charger pushed in 2870mAh to full charge. The error was 5% which is excellent for a low cost device and good enough to rely on for maximum safe flying time.

Rpm
I think the rpm is wrong. According to the spec it should be 14.4 x 670 = 9648. I will check that I have the number of coils correct for the RPM sensor. The spec gives 12 pole but is this the figure that should go into the sensor edit? The full throttle readings were 4660 static and 4875 in the air. This is about half what it should be. The numbers give about a 5% unloading in the air. I edited the sensor to 6 pairs rather than 12 poles and got a reading of just over 9000.

Power
After carrying out the flight tests I realised that I could have recorded power as well. I then created a calculated sensor called **Watt** by multiplying current and voltage and setting the unit to watt W. I selected integer value so avoiding decimal places. I displayed Watt and Watt+ on the screen. I carried out a static test. Clearly the current is measured in amps not milliamps as the simple multiplication gave watts. With a fully charged battery I got a reading of just over 900W. The full charge voltage is 16.8 so multiplying this by 55A gives 924W. Therefore the calculation seems to give a correct result. It also shows that under full charge the motor is being asked to produce slightly more than the specified power.

Conclusion
The propeller is about right, though it is pushing the motor to its current limits. The telemetry gives good results.

Sources

Ebay sources for devices

S.Port Air Link-S	Megafun2016 eModele.net (Cracow, Poland) £22.87 inc postage or Banggood £29.11 inc postage
BLHeli toolstick	rc-terminal (Erfut, Germany) £16.01 inc postage

Other

Neurons	T9HobbySport : https://www.t9hobbysport.com.

Software and manuals

BLHeliSuite32:
http://www.mediafire.com/file/9ylhp0s0ces4tw2/BLHeliSuite32_32700.zip/file
Freelink app: https://www.frsky-rc.com/s-port-airlink/

S.Port Airlink

This USB device can be used to upgrade the firmware, and to configure, telemetry devices and receivers that include telemetry such as S6R and S8R. This is work in progress and I will add to this section when I know more.

Change log for this manual (from ver 1.4)

1.4 900 MHz equipment and use described
1.4.1 Problems with sound buzzing and **Play value Cels** removed. Full list of telemetry sensors added. This change log added (17 Nov 2018)
1.4.2 Changes and additions to Glossary and small textual improvements. Improvements to buddy box instructions. Added note about vario and pressure changes.
1.4.3 Note added about use with a flight simulator. Note about live changes added.
1.4.4 Instructions for using a flight simulator 'wireless' dongle added. A few minor text changes.
1.4.5 Air speed indicator added and example of installation in Bixler. Note added about changing telemetry device numbers and nature of the telemetry connection. Added how to set vario sink tones off. Changed default style settings (note for me only).
1.4.6 (Jan 19) Much more technical stuff on telemetry added as a separate page at the end of the manual just above this change log. Data logging added. Current sensor use added. Instructions on using **Consumpt** added. Test data on 40A Consumpt data added. Battery monitor added to full house.
1.4.7 (Feb 19) Info about airspeed indicator testing added. Flight tests on 40A sensor and consumpt added. Improved the instructions for dual rate.
1.4.8 (May 19) Flight test data for 150A current sensor added
1.4.9 (July 19) Neuron ESCs added
1.4.10 (Sept 19) More on Neuron ESCs. Opening note about manuals. A few other text improvements.
1.4.11 (Sept 19) Much more on Neurons. Airlink and toolstick added. Test flight results added.
1.4.12 (Jan20) Neuron S versions added
1.4.13 (Jul 20) Some typos corrected
1.4.14 (Aug 20) Keith Eldred's method for setting flight simulator channels added.
1.4.15 (Nov 20) Minor textual improvements and servo power board picture added.
1.4.16 (May 21) Warning about fragility of Neuron Ss. Trimming in buddy mode.
1.4.17 (July 21) PicaSim flight simulator added.
1.4.18 (Aug 21) Use of curves for the Hobby King paramotor
1.4.19 (Aug 21) Picasim improved and contents table moved.
1.4.20 (Aug21) Creating a new model changed so it suits beginners
1.4.21 (Aug 21) Beginner's bit moved to start
1.4.22 (Sep 21) Adding receiver redundancy
1.4.23 (Nov 21) More detail added about telemetry screens
1.4.24 (Jan 22) Calibration for Neuron 60 added
1.4.25 (Jan 22) How to test your transmitter aerial
1.4.26 (Jan 22) Added info about ACCST V2
1.4.27 (Feb 22) Info about 900 MHz and FHSS added
1.4.28 (Feb 22) Repairing the aerial added.
1.4.29 (Feb 22) Warning about legality and interference of 900 MHz and extra on Lua
1.4.30 (Mar 22) Added standard telemetry screen design
1.4.31 (Mar 22) Repairing a receiver aerial and additions to glossary
1.4.32 (Mar 22) Corrections to list of audio messages
1.4.33 (May 22) Some text errors corrected Change to test bed. Changes for RCSD.
1.4.34 (Feb 24) Added section on curves and throttle curve