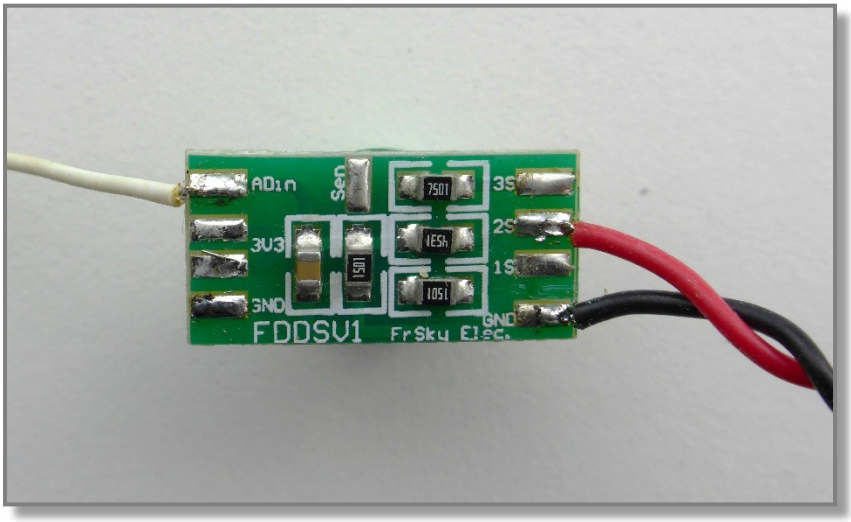
How To ... Part 2

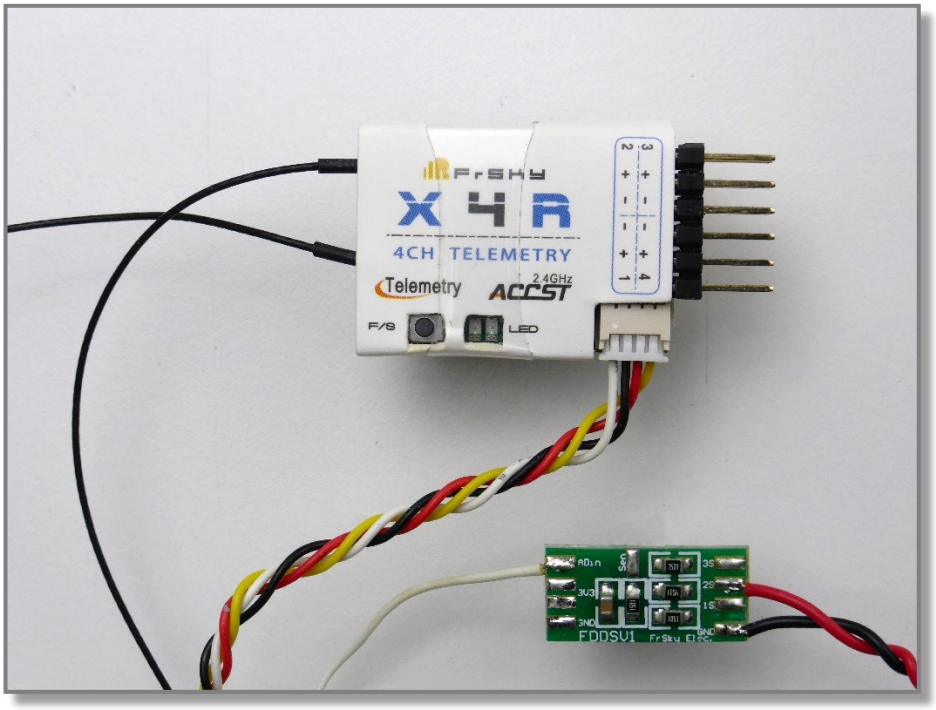
Contents

How to Add Voltage Telemetry to the X4R

The X4R is a small lightweight receiver with 3 or 4 channels depending on the model designed for small light models. Often in such planes, space is at a premium and it is difficult to fit any telemetry in. With small foamy electric models, it would be very useful to have at least voltage telemetry to give an indication of battery voltage. This could be very useful with powered gliders where there is a mixture of powered flight and gliding, and using flight times alone is not appropriate. The X4R receivers come Smart Port enabled so that the standard voltage sensor could be used. However, this is nearly the same size as the receiver itself. Another feature of these receivers is, unlike the other X series receivers, there is an analogue input, thus some of the older, pre-Smart Port, sensors can be connected. One is the **FrSky FBVS-01** battery voltage sensor. This tiny sensor is ideal for measuring flight battery voltage and is very inexpensive, although it is not always available from the main suppliers. However, a quick internet search will soon pull up a few places that sell it.



Fitting the sensor is quite straight forward. With each X4R comes a 4 way plug and lead to fit the smart port socket. Three wires are for the smart port and consist of a red positive, a black negative, a green or yellow smart port lead, and a white wire not connected at the other end. I firstly fit a normal servo connector onto the red, white and green/yellow lead to have a smart port plug just in case I want to fit other sensors or update the firmware.



First, remove the heat shrink from the sensor. Don't worry, a spare piece of heat shrink is provided. Then unsolder the red and black from the **GND** and **ADin** pads. Then solder the white lead onto the **ADin** pad at the top left-hand corner. The red and black leads soldered to **GND** and 3S can be left as they are. Only one negative lead is required as the two **GND** pads are linked on the circuit board. If using a 3S battery connect the red wire to the 2S terminal instead to get a somewhat more accurate reading. Don't worry, I cannot understand the logic either. Then the red and black leads are soldered to the power connector going to the speed controller.

Power up the transmitter and receiver and set **OpenTX** to search for new sensors. Now **A2** should come up as a new sensor and show a 13.2 volt ratio. Check the actual battery voltage and then adjust the offset until the telemetry reads the same as the actual battery voltage. Finally, once everything is working, replace the heatshrink with the spare length of clear heatshrink provided.

The X4R receiver analogue input has a maximum input of 3.3 volts. The sensor has three terminals:

1S terminal which measures voltages from 0-6.6v
2S terminal which measures voltages from 0-13.2v
3S terminal which measures voltages from 0-19.8v

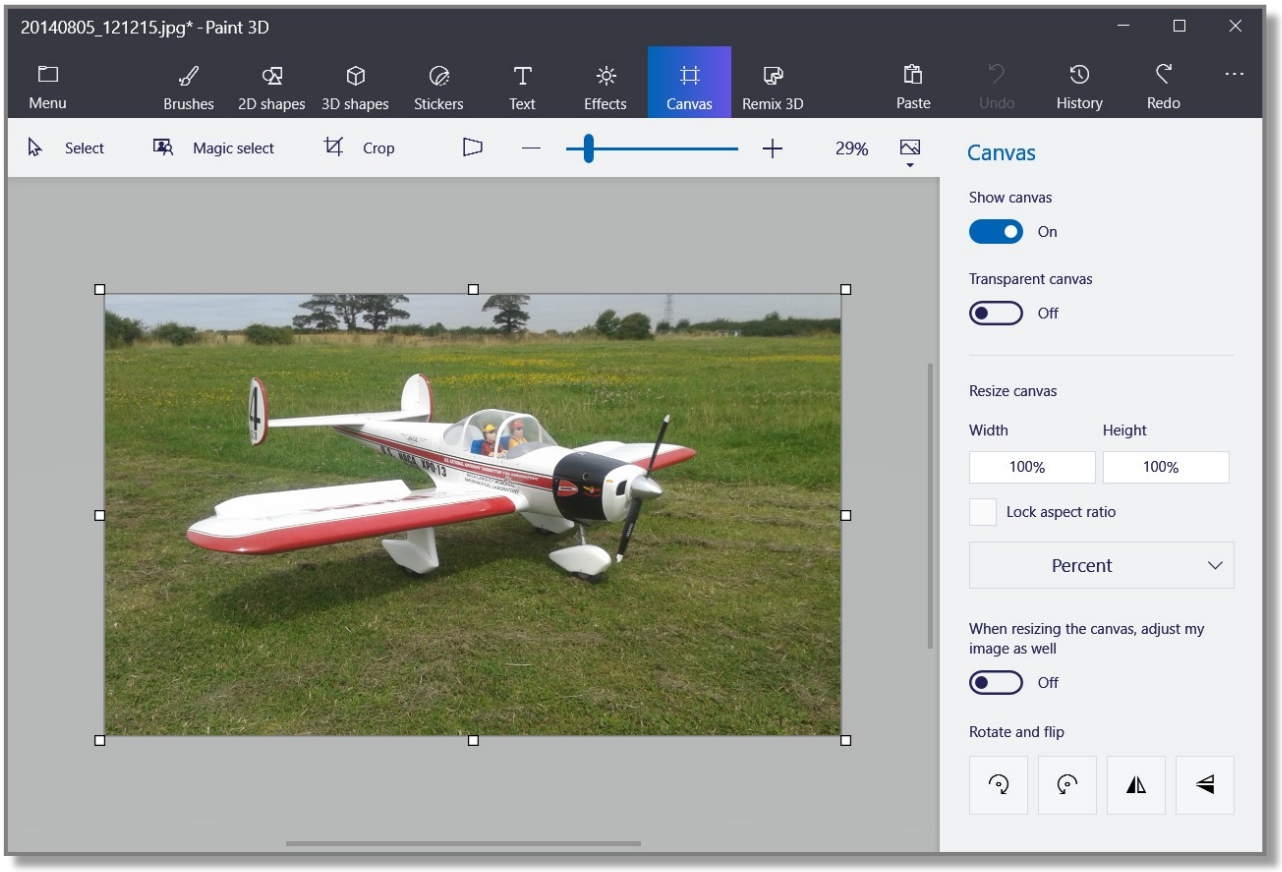
If using a different battery or using the 1S or 3S terminals, then the ratio on the **OpenTX** telemetry page will need adjusting to match the above maximum voltage.

How to Create Model Pictures for the Horus

There is a good collection of Horus model image files at this website:

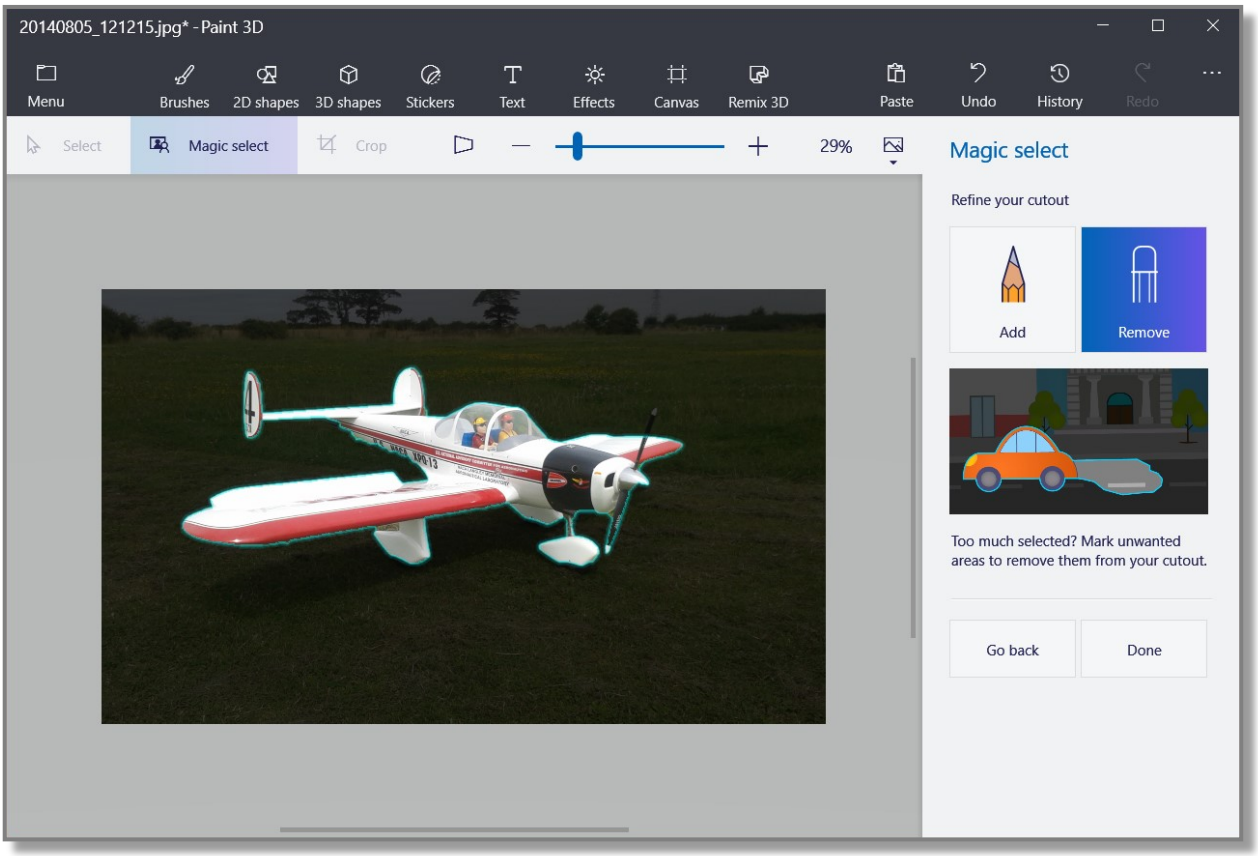
https://skyraccoon.com/icons

For one's own unique models, it is useful to create one's own images for the transmitter. A good drawing program is needed to be able to modify photographs or pictures to make them suitable for the Horus. Perhaps the best for this need is a program called **Paint 3D** from Microsoft. This free program is ideal for this sort of work, and is the program we will be using here. For a standard model, the advertising picture can be used and adapted. For one's own models, it is possible to use a photograph:

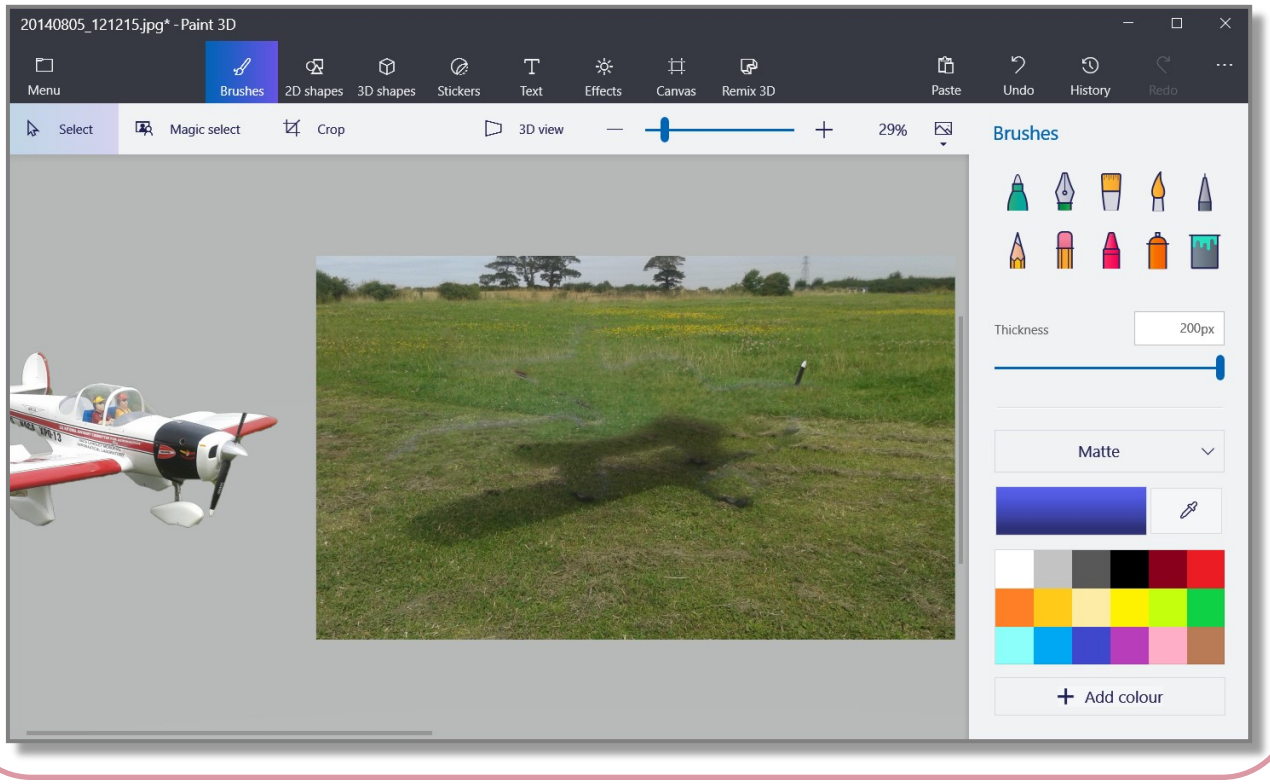


As can be seen, this is a good photograph of the model, however the background is not so good!

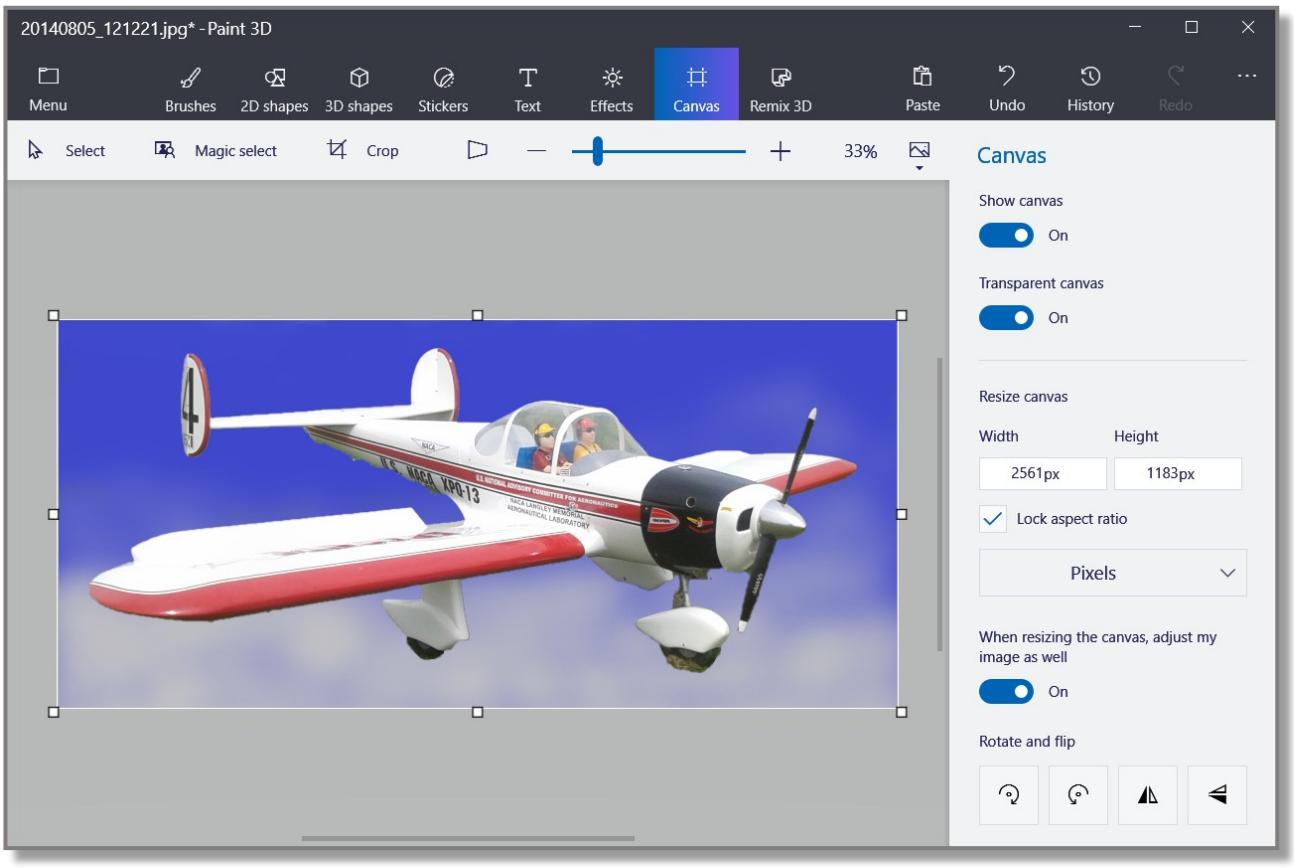
Use the **Magic Select** function show on the toolbar and use the **Add** and **Remove** options to tidy up the cutout and click **Done** when ready.



Click on the model to be cut out, and slide it out of the way.

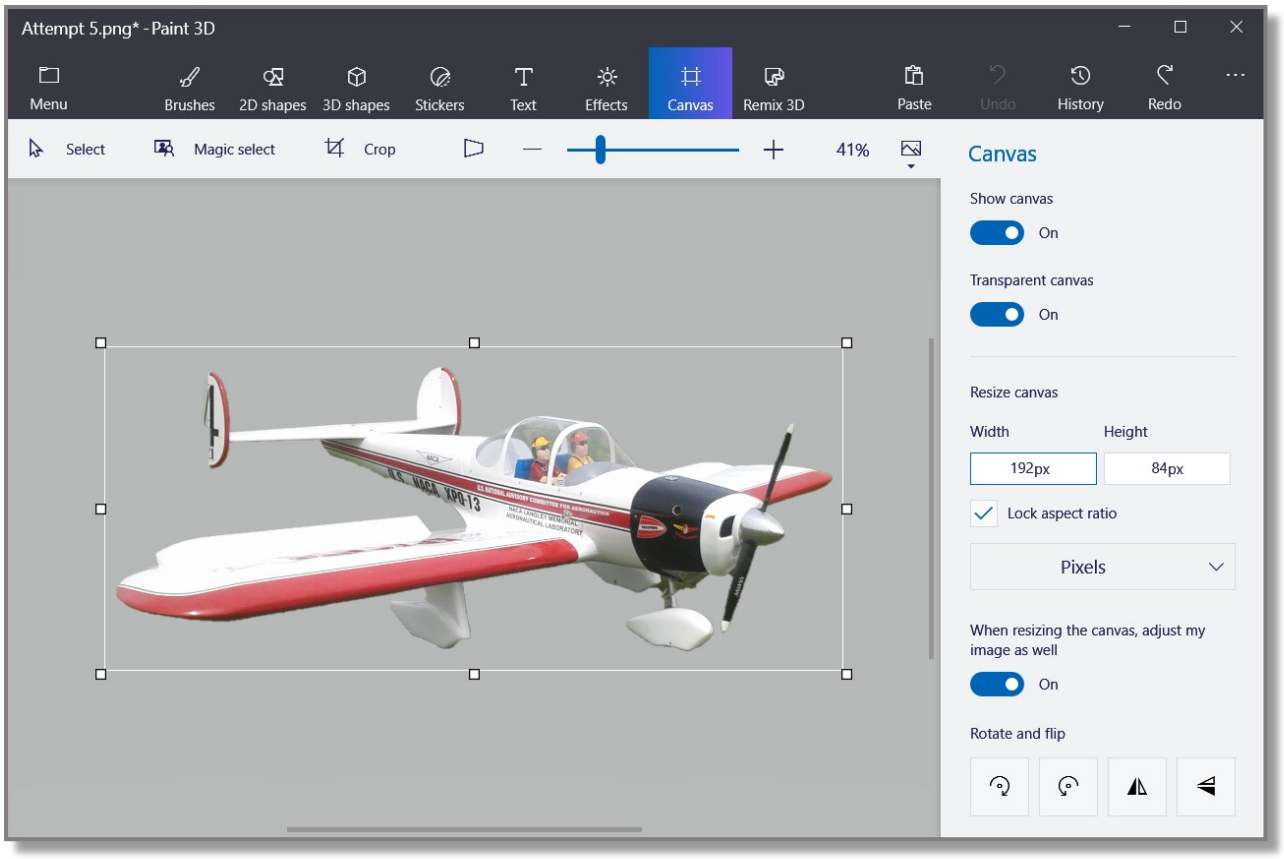


Now recolour the background to suit the model. This can be done with any of the brush tools. As this example model is white, a blue background has been created, and then a light grey colour has been sprayed over part to give a sky effect.



Then slide the cutout back over the background. Finally export the picture as a **png** or **jpg** file. Now reload the saved picture and resize for the Horus screen. This should be a maximum of 192 pixels by 116 pixels. For some reason this only works if the resizing is done after being reloaded. First lock the aspect ratio, then adjust either the width or the height to get the largest picture. Finally resave, remembering to make the filename no more than 6 characters long.

If the above screenshot is compared to the one on the previous page, it can be seen that this image has a transparent background. This is quite easy to achieve with **Paint 3D**. Once the **Magic Select** has been used to cut out the model, slide it out of the way and use the rubber to erase the background. Then restore the model back onto the background and **Export**. Next reload it, crop to size and save as a reduced size file ensuring the **Transparent Canvas** is enabled:



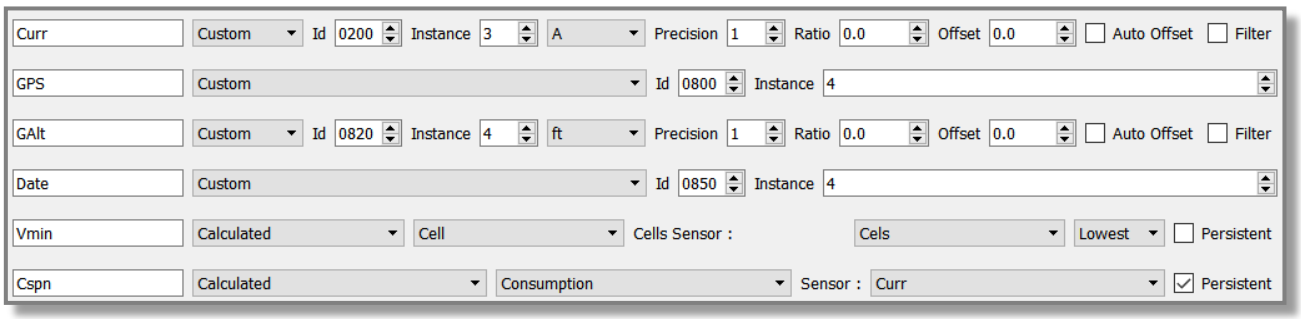
How to Create a LiPo Fuel Gauge

This is a useful procedure to display a telemetry setting as a percentage. It was devised by Mike Naylor (Miami Mike) and is reproduced here with his permission.

This is a "fuel gauge" setup for **OpenTX** to display the flight battery's state of charge as a percentage, beginning at 100% for a fully-charged pack and decreasing as the battery consumes power. It requires a current sensor, such as a **FrSky SP-40A - Smart Port 40 Amp Sensor** or **FrSky SP-150A - Smart Port 150 Amp Sensor**.

It allows alarms and voice announcements to be added that trigger at specific charge states, or at regular time intervals, or each time the charge state decreases by a certain amount so that there will be announcements at, for example, 100%, 90%, 80%, etc. Announcements can also be brought up at a flip of a switch. The battery state will be retained between flights even if the radio is turned off, so a switch is provided to reset the charge state to 100% when the battery is replaced or recharged.

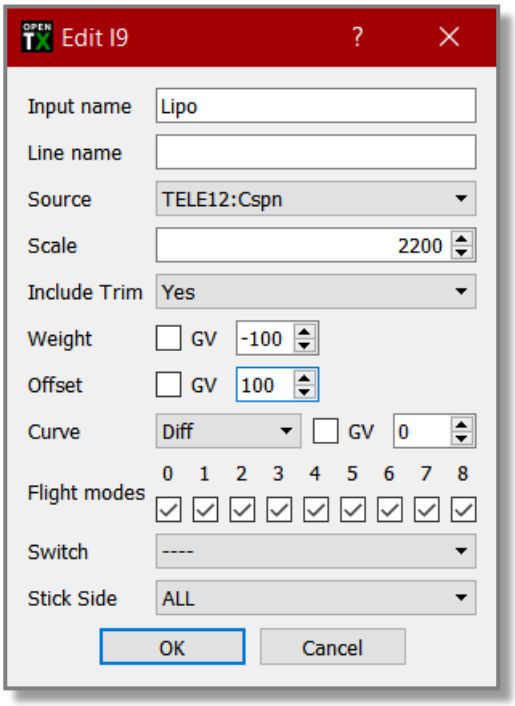
This idea could also be adapted to give percentage readings of other telemetry values.



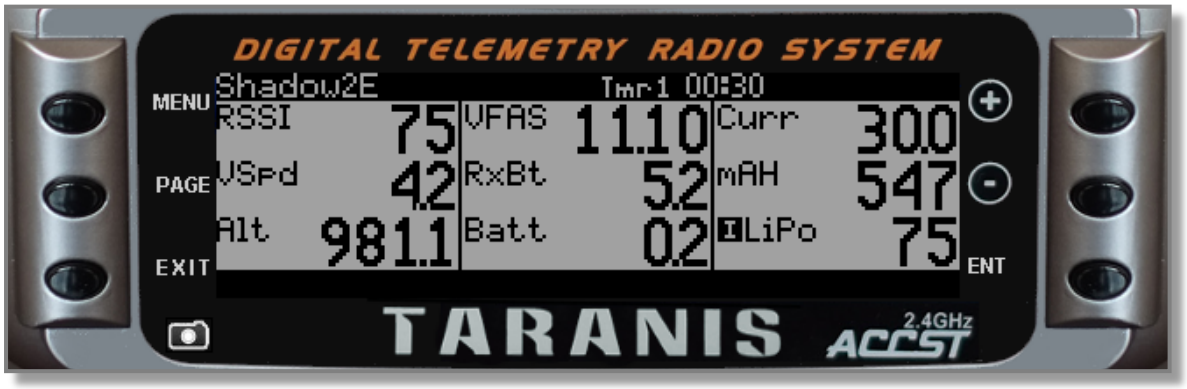
First the current sensor needs to be discovered, and then a new sensor added to calculate the consumption. Here it is called **Cspn**. Notice the **Persistent** box is checked to retain the value if the transmitter is switched off.

Now for the clever bit. A new input is created on the **Inputs** page. Use a channel above the normal channels used for a model. In the example below, channel 9 is used.

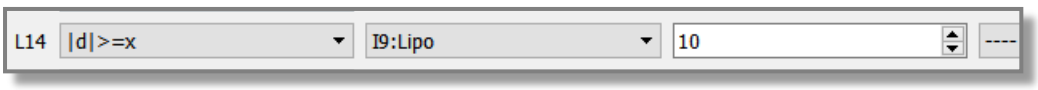
The Weight is set to -100 and the Offset is set to 100 to give a range of 0% to 100%. The Scale is set here to 2200, which is the capacity of the battery. In the scale setting, the capacity of the battery is entered to give us an accurate percentage for the battery being used. One could decide that 0% is when the battery is fully discharged and then only fly whilst the battery is above 20% say. On the other hand one could set this figure to 1760 so that 0% would now represent the minimum flying time, and still leave the battery 20% charged.



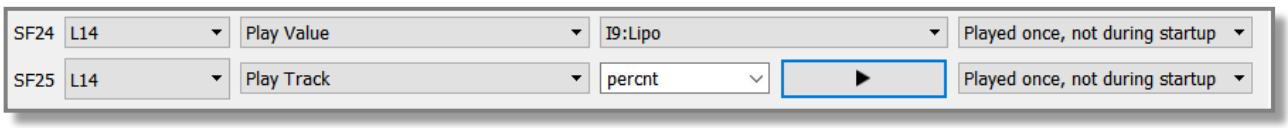
On the telemetry page of the transmitter, an input can be selected as well as the more normal telemetry figures. Thus the percentage remaining of the battery charge can be displayed. Here the value of 75 is displayed in the bottom right hand corner of the screen.



To add an automatic call down every 10%, first create a **Logical Switch**:

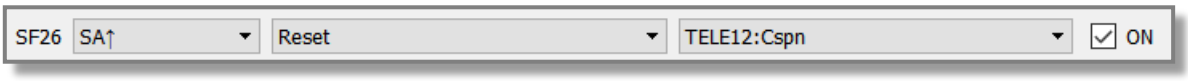


Then add two **Special Functions** to enable the announcement.



There is no percent sound in the user sound files, however, a percent is available in the **SYSTEM** sub folder of the **SOUNDS** folder on the SD Card. It is called **Percent0**. This needs copying across

to the main **SOUNDS** folder and the name needs shortening to just 6 characters. While generally the **Special Functions** can be entered in any order, in this example, as the routine uses the same **Logical Switch**, the one that appears first on the list will be executed first. A switch could also be set up to reset the **Cspn** value:



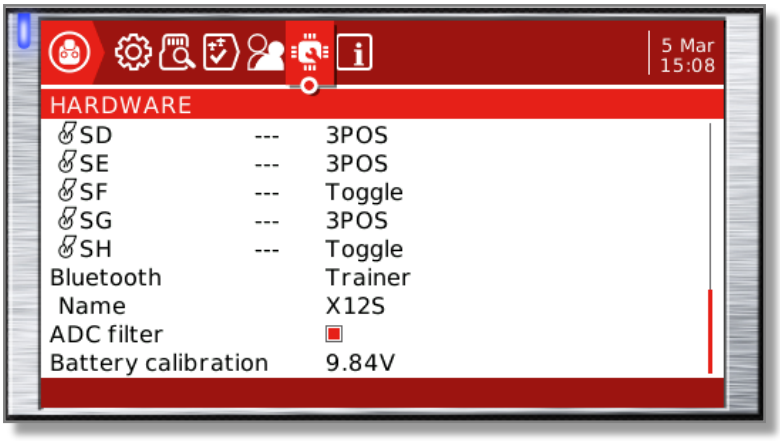
If the battery is normally changed each flight, then this switch would not be needed, and the **Persistent** box should be unchecked on the **Telemetry** page.

No doubt readers will be able to think up all sorts of uses for this routine. The clever bit is using an **Input** to store a particular value in the **Scale** parameter and use the **Weight** and **Offset** to give a range of 100.

How to Use the Bluetooth Trainer Function

The Horus X12S and the Horus X10 and X10S transmitters all have Bluetooth facilities which can be used for telemetry and trainer functions. This **How To** shows how to set up the Bluetooth to work with the trainer function.

Setting up the Bluetooth trainer must be done on the transmitters, NOT the **Companion**.



1. Go to the **Hardware** screen of the **System Menu**. Scroll down to the Bluetooth settings, and enable **Trainer** mode and then give the device a name. It would appear this is not used for the trainer function.
2. Do the same for the second transmitter being used.
3. Set the **Slave** transmitter trainer function to **Slave/Bluetooth** in the **Model Setup** menu.
4. Set the **Master** transmitter trainer function to **Master/Bluetooth** in the **Model Setup** menu.
5. Now on the **Master** transmitter, the box below, the **Init** box should change to **Discover**. Click on the **Discover** button, and both transmitters should pair.