12-channel reed encoder "Reeduino" with S/C emulation

I've done an arduino reeds emulation encoder for up to 12 channels that suits any of the 'ebay boards' including the small one with the inbuilt 328P processor. I've deliberately worked to the spec of the existing PIC 12ch reeds encoder, and other than a few bells & whistles, its functionally identical to the old PIC encoder.

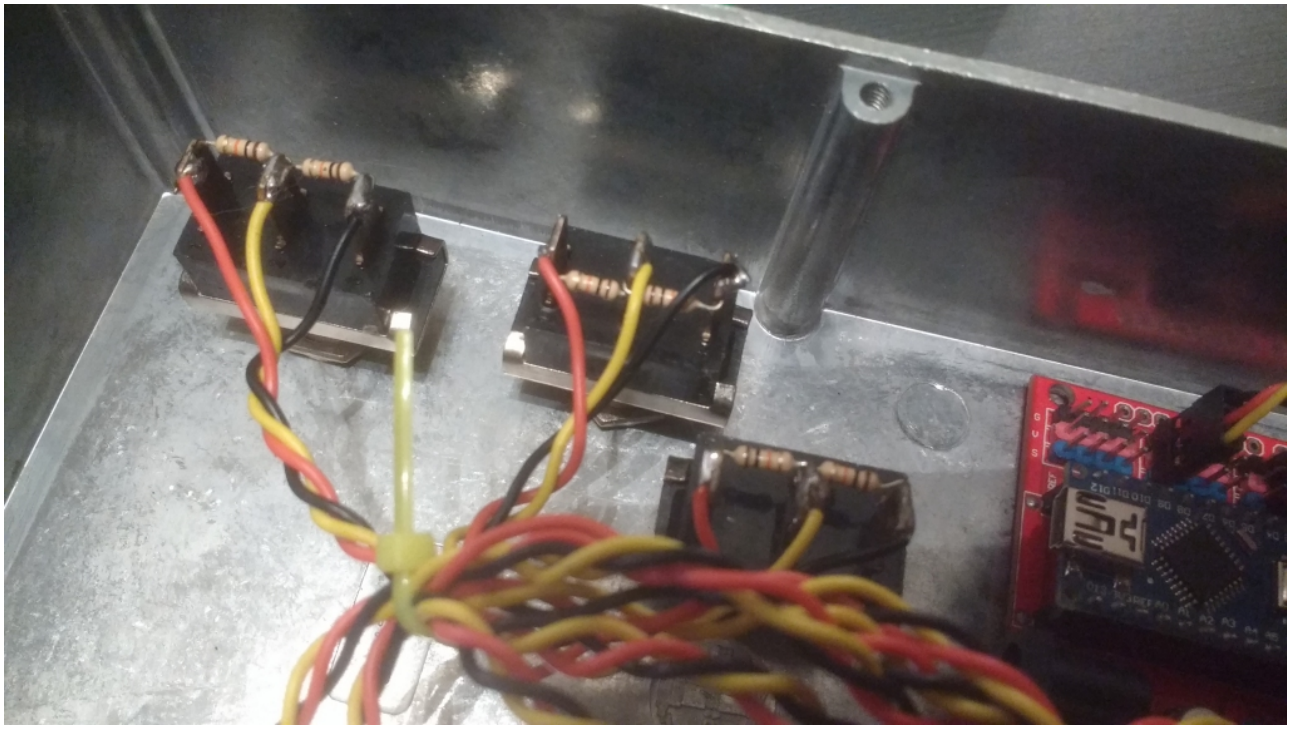
In use it will be totally familiar to anyone who's used to the old PIC reeds boards – it remains an accurate portrayal of a period reeds set which is easy to assemble and dead simple to use.It has all the same facilities of the old one, reversing by power-on with the toggle thrown (saved to flash), variable servo-slow via a pot, half-speed throttle & aux channels, proper elevator trim via the trim toggle and 'cheating trims' (with pips) on all the other channels by simultaneously holding elevator trim and the required channel to trim, range-test sweep, etc – plus a few extras such as ATV (adjustable travel volume), the Single-Channel escapement emulation mix and the V-tail mix – but otherwise its exactly like the old one in every respect. Its wired the same way too, with resistors across the toggles to enable three positions to be read via one analogue pin - and as before the actual analogue value has no direct bearing on the servo, its quantised to just 3 values so we can tell if the toggle is pushed one way, pushed the other way, or in the middle. There are crude reed implementations out there where resistors hold a propo channel at neutral and simply switch to either travel extreme - sometimes with capacitors for servo-slow - but thats not the case here. "High - mid - low" is just an economical method of reading the three possible switch positions via just one arduino input.

So if its exactly the same, why bother? Well the old PIC reeds encoder boards were expensive, even the bare PCBs cost me £6, then theres a boardful of components and each one took me a full evening to build & test. I always felt guilty that I had to ask £25 to cover costs & delivery but I just cant do them for less.
The 328P board on the other hand is complete and readily available for £3 or thereabouts, and it becomes an easy DIY project. Easier and cheaper, and with a few nice extras.

Oh dont forget that A4 & A5 are switched around on the little 328P board, thats the aux & trim toggles in this particular case.

I will do a video when time permits, not that theres anything new to see - apart from the S/C button you wouldnt know its not the old PIC encoder, which was the intention all along.

Assembly:
First prepare the toggle switches by soldering 10k resistors between either 'make' contact and the common.
All the toggles are wired with servo plugs, pos and neg to the 'make' contacts, and signal to the common.



The single-channel button can be omitted if its not required. One button contact is ground, and the other can be wired

either directly to D12 for 'sequential', or to D11 for 'compound' operation. A change-over switch can also be used to connect the button to either D10 or D11, this gives a choice which can be changed in flight.

The V-tail mixer can also be omitted if its not required. One contact of the switch is ground, the other is wired to D7. Use a servo lead but omit the pos. If the V-tail controls are wrong, try the usual variations, working down the list in turn until its right:

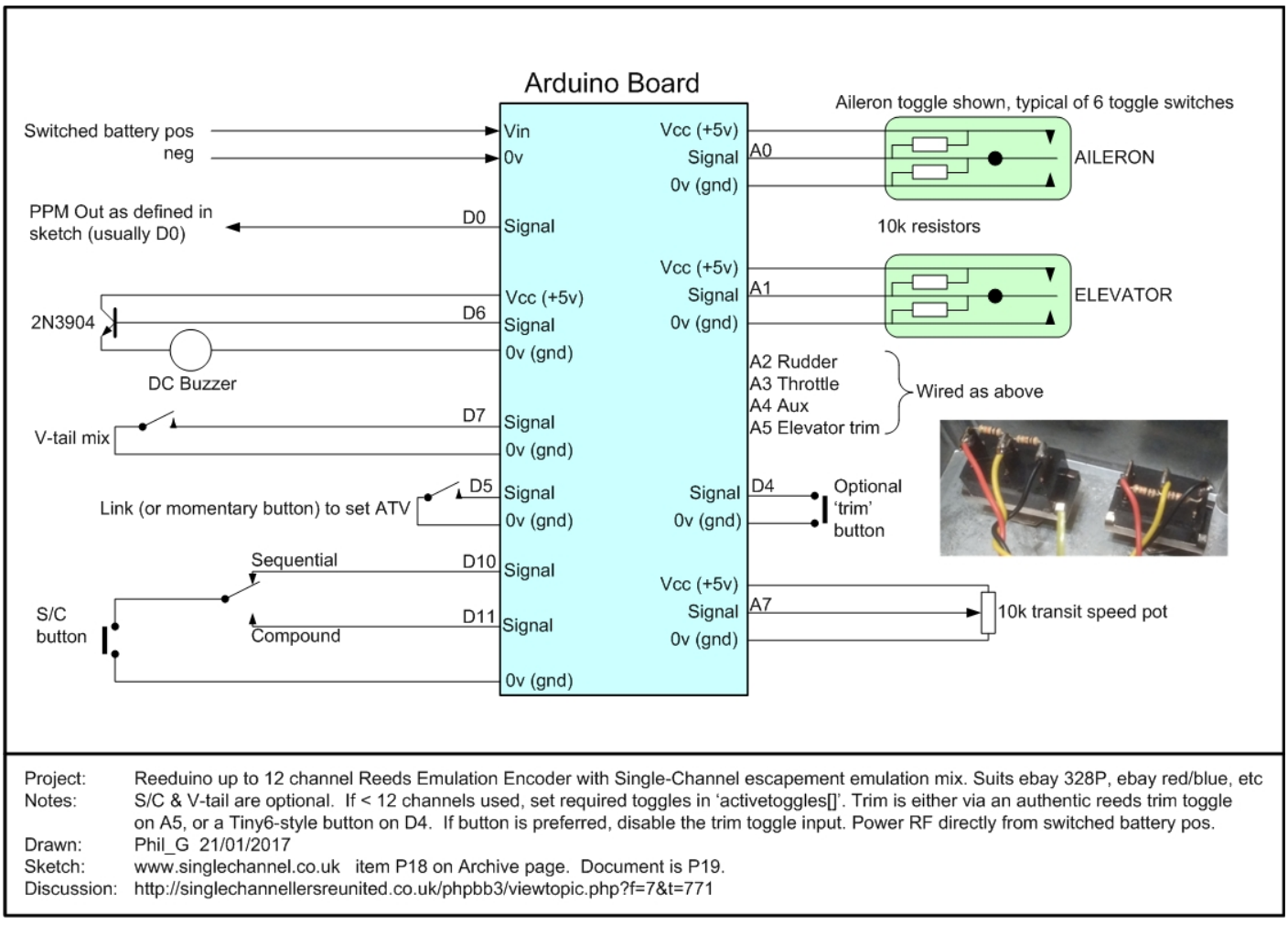
1. Clear all reverse settings & try the controls.
2. Reverse the first servo
3. Clear reversing on the first and reverse the second servo
4. Reverse both servos
5. Swap the two servo plugs in the receiver and go through the list again. One option will be right!

The buzzer on D6 is best buffered using a transistor. This is most easily done on the buzzer itself – see addendum. Please dont omit the buzzer – its unobtrusive and trims would be awkward without it.

The transit-speed pot is not optional, it is essential to set and hold the required speed.

Whilst its an 'up to 12 channel reeds' encoder, some projects dont need the full 12 channels and you can't just leave unused toggles disconnected as the inputs are then indeterminate as they float around picking up extraneous voltages. So we need to enable the toggles you're using, and disable the ones we're not. This is done with one line of configuration by editing the 'activetoggles' array which is near the top of the sketch. Note that this setting *must* match your hardware – for example, if you ask me to supply a 12 channel encoder, you *must* wire all 6 toggle inputs. Remember that in a RET model you're using the aileron channel for rudder. Think 'primary steering control' .

The PPM output pin is defined in the sketch – I use D0 for convenience as its available on the 328P board end header. On the red/ & blue Nano boards we've used D13. Remember Frsky V8 boards need protection on the PPM input – either a schottly (band towards encoder) or 2k resistor – other modules are fine without.



Project:	Reeduino up to 12 channel Reeds Emulation Encoder with Single-Channel escapement emulation mix. Suits ebay 328P, ebay red/blue, etc
Notes:	S/C & V-tail are optional. If < 12 channels used, set required toggles in 'activetoggles[]'. Trim is either via an authentic reeds trim toggle on A5, or a Tiny6-style button on D4. If button is preferred, disable the trim toggle input. Power RF directly from switched battery pos.
Drawn:	Phil_G 21/01/2017
Sketch:	www.singlechannel.co.uk item P18 on Archive page. Document is P19.
Discussion:	http://singlechannellersreunited.co.uk/phpbb3/viewtopic.php?f=7&t=771

Trim options:

There are two options for trim, both of which permit you to trim all channels, unlike a genuine reeds set where only elevator trim would be available.

1) a Tiny6-style trim-button which will be totally familiar to anyone presently using a Tiny6.
2) A conventional elevator-trim toggle

Without any code changes you have the choice of either the realistic reeds trim toggle, exactly like a period reeds set, OR the Tiny6 trim button, whichever you prefer. (actually you could have both but thats totally unnecessary! …)

The trim button is a hardware option and can be omitted if you prefer the authentic up/down trim toggle, but it does simplify a basic Rudder/Elevator/Throttle 6-channel reed set as you no longer need an extra toggle for the elevator (and other channels) trims.

If you're building a full-house set, for authenticity I'd suggest fitting all the toggles (omitting 'aux' for a 10-channel set). Dont fit the trim button, and nothing has changed - its the familiar full-house reeds layout. Elevator trim works as you would expect, just like a period reeds set, and using a simultaneous combination of trim and other channels, they can be trimmed too.

If however you're building a simple 6 channel RET set, I'd use the button rather than have another toggle switch which would spoil the appearance of a 6-channel set, as traditionally they would have only three toggles. If you do omit the trim toggle, set it as inactive in the 'activetoggles[]' array - ie the last entry in activetoggles[] will be a zero. There are examples in the header - eg a Tiny6 clone would use {1,1,0,1,0,0} for three toggles, aileron, elevator & throttle. These three toggles would physically connect to A0, A1 & A3 respectively: (aileron is your primary steering channel in a RET configuration)

```
// Configuration section:
// indicate which toggles are wired up, 0 for no, 1 for yes in the order: aileron, elevator, rudder, throttle, aux, trim
const bool activetoggles[] = {1, 1, 0, 1, 0, 0}; // 1 for the toggles you are using, 0 for unused ones, this prevents unwired ones reading randomly
```

To trim using the button, simply hold it in and blip the controls in the direction you need trim. Release when done.

The trim button can be quite discreet, as on my own RCS Inter-6: (between the RCS logo and the 'Inter 6' decal)

Transit-speed pot:

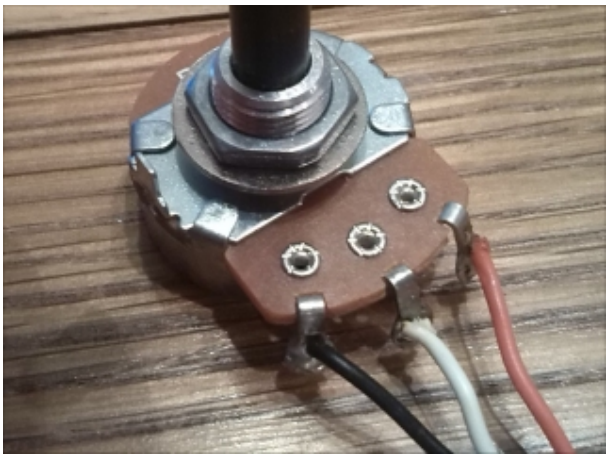
Typical period reed servos such as the Bonner Duramite had a transit speed of slightly over 1 second lock to lock. Climax ServoMites were a little faster, as were Graupner Bellamatics.

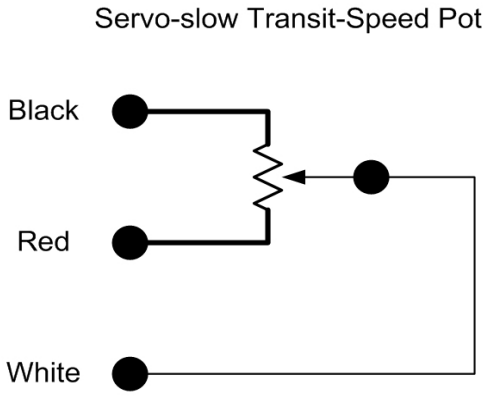
The technique for flying reeds smoothly is to pulse for gentle control, and servos which were too fast did not give the desired proportional effect.

With gentle pulsing, a Duramite would almost hover at mid travel, giving a gentle turn.
So whereas for conventional proportional flying, fast transit servos are desirable, for reeds we need a slower transit to enable the servo to average out the pulsing of the toggle. This pot adjusts the servo transit speed.
For ease of throttle setting, the throttle and aux channels run at half the selected transit speed.
The speed takes account of ATV – a servo with half as far to travel would otherwise get there in half the time!

The pot is a potential divider so its value isnt critical, 5k, 10k, 20k will be fine. Transit speed isnt a feature you would normally change in flight, so the pot can be mounted internally and set to personal preference – it could even be a preset.

The pot is wired with 5v and ground across the two outer tags and the wiper (centre tag) via the signal wire to A7:





ATV Adjustable Travel Volume

Adjustable Travel Volume or ATV as Hitec call it, its a settable end-point for each servo.
Its not variable 'in flight' like rates on a propo set - thats not what its for - its a configuration setting for use on the ground when setting up a new model, to allow you to pre-set the maximum servo travel & hence control-surface movements on each channel. ATV also affects the single-channel emulation throws.

To change ATV, you switch on, check everything is working, then stick a Spekky bind-plug into D5 or press the ATV button if you've provided one. This grounds the D5 signal wire. It beeps 'A' for ATV.

Then, two things happen.
First, all channels become progressive, ie they do not self-neutralise but stay where you left them like the throttle & aux, and second - regardless of the pot setting the servo transit speed is temporarily set to its slowest setting.

This allows you to slowly nudge any or all of the servos to whatever extreme limit of movement you require. Say you want 1/4" or aileron movement, you nudge the aileron toggle which is now progressive, left & right until the aileron is up or down by exactly 1/4". Same for elevator, rudder, throttle & aux.
Say for example you have an IC engine and the throttle arm has only one hole, and the servo also has only one hole. Without ATV the servo might hit the carb stop and stall. With ATV you can set the end point so the carb closes perfectly without doing any mechanical adjustments.

These adjustments can be nudged, twiddled & repeated as long as you like until all the throws are where you want them.

Then, when you're happy, you pull the link off D5. It saves the settings to flash, beeps 'V' for 'end of ATV', the servos neutralise and everything reverts to normal operation using your new servo throws. Any toggles you dont touch will retain their previous ATV setting. Any toggles that you try to set below 20% throw will also revert back to their previous ATV setting.

Thereafter, when you switch on, and it reads your reversing and trim settings from flash, it will also read the ATV settings you just made.

The saved servo-throw limits and are kept until you want to make changes by doing the 'link on D5' thing again.

Servo transit speed is proportional to the set ATV for each channel, this means end-to-end transit time is the same regardless of how far each servo moves under its ATV. The reason this is necessary is because with a constant speed setting, a servo that had half as far to move would get there twice as fast.

In practise the servo transit pot works exactly as before although strictly it controls transit-time over all the different channel ATV's, rather than rotational speed.

I've limited ATV to 20% minimum so you cant accidentally set 'no travel' which would be a bit exciting.

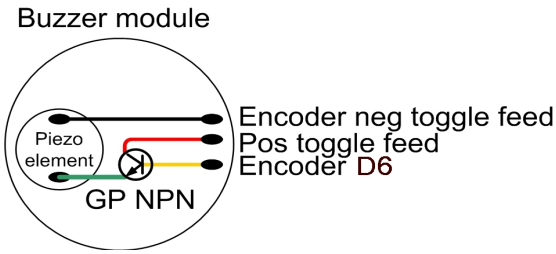
As before the flash reset command (power on with up-trim held) clears all reversing and centralises all the trims but also now resets all the ATV's to 100%, which is 1100 to 1900uS, plus or minus 100uS total trim.
The ATV range is from 20% to 112.5%

Doug will be happy to hear that throttle is treated exactly like any other channel, with reversing & ATV, but as before is slowed by half with respect to the other channels to make throttle selection easier.

Because an ESC 'trains' itself to the throttle channel, there should never be a need to change throttle ATV on an electric, and since almost all ESCS are 1mS low, 2mS high, there should never be any need to reverse an electric throttle, but in any case please, please remove electric props before doing any throttle reversing or ATV settings!

If you dont want ATV, just ignore any reference to it and the set will happily assume an invisible 100%.

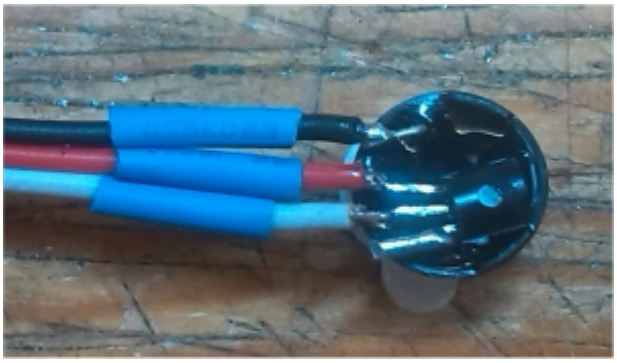
Buzzer:
The buzzer is used for trim-pips, mode selection and for the inactivity warning. It is normally a two-wire device but in this case a transistor has been added to buffer the buzzer drive. The whole thing as a unit is supplied pre-wired as follows and can be considered a 3-wire device as per the previous diagrams:



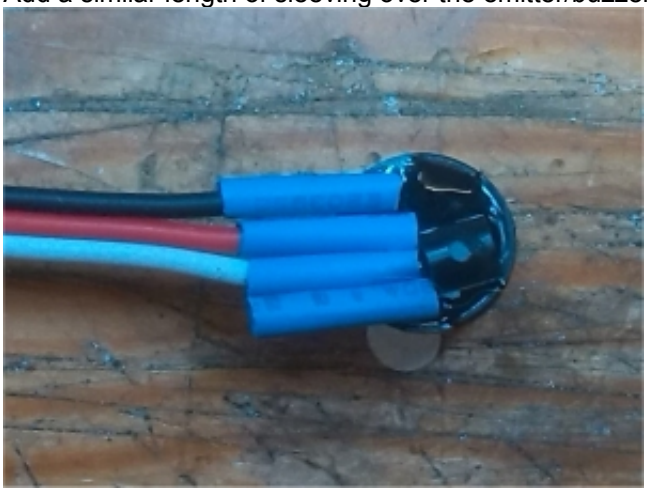
The buzzer assembly (buzzer and transistor buffer) is supplied ready made & tested, but here's how I assemble them, you may come up with something better:

First, with the buzzer positive at the bottom, the leads are kept parallel whilst being bent to the left along the surface of the buzzer:

Next the 2N3904 transistor is laid flatside-down with its emitter touching the buzzer positive, then soldered:
The leads are cropped to the edge of the buzzer and sleeved servo cable soldered as follows: black to the buzzer neg, red to the collector, white to the base. Theres no wire to the buzzer negative.



Add a similar length of sleeving over the emitter/buzzer negative:

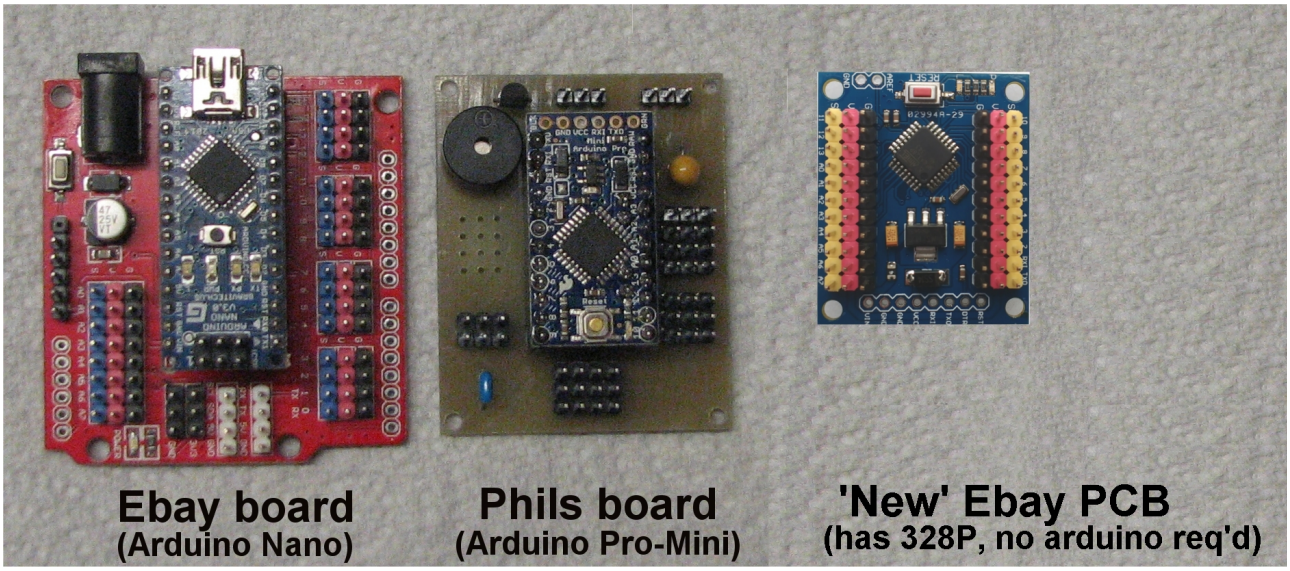


Next heat-shrink the sleeving, making sure it remains pushed up against the transistor body. Then add an overall sleeve covering all the connections:

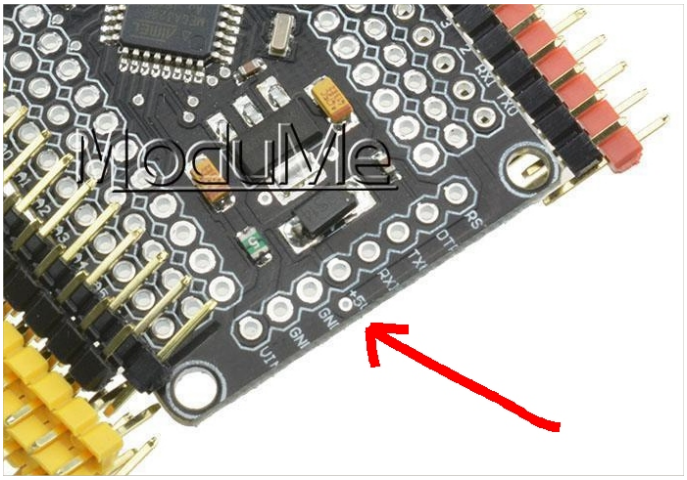
When connecting the assembled buzzer module, red is positive 5v (from the toggle switches positive feed) black is negative (again from the toggle feeds) and white goes to D6 on the encoder.

Arduino PCB notes:

We've found some very convenient PCB's on ebay which are cheaper than component costs. Here they are compared to my own PCB:



Recently a variation of the 328P board has been supplied, for us its a better board and whilst at a quick glance it looks identical, its easily identified by the tiny Through-Plated-Hole (TPH) adjacent to the Vcc (+5v) pin:

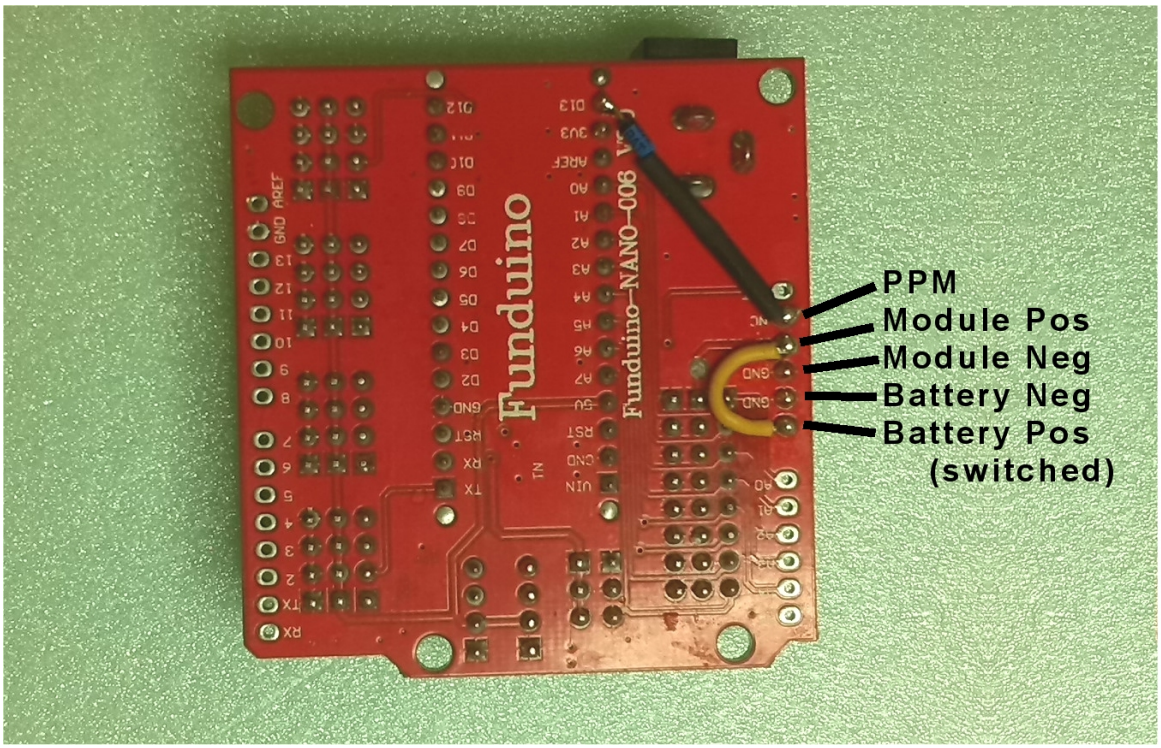


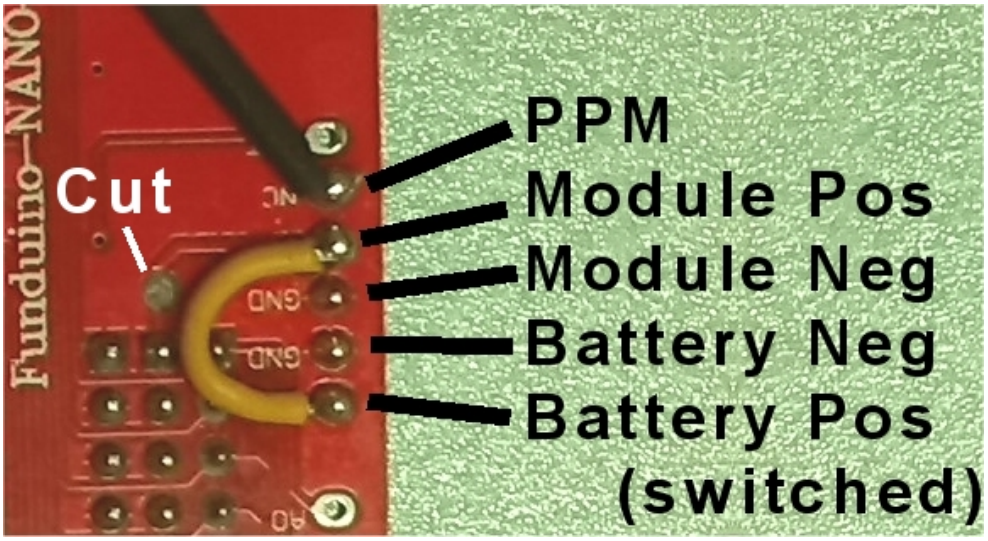
The first became known as the 'red ebay board, though it was also available in blue. It has a socket for an Arduino Nano. It needs a couple of simple mods to convert to an encoder board.

The ebay PCB on the right is a smaller, neater board with an ATMega 328P built in, so no separate Arduino chip is required. We refer to this one as the 'new 328P board'. It also needs a couple of simple mods.

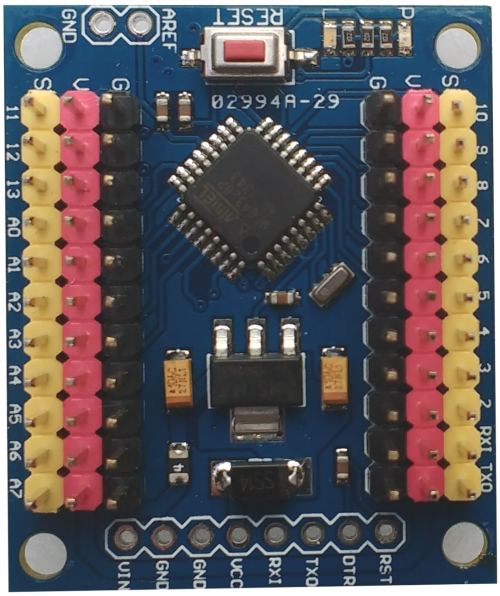
The boards need all their supplied headers fitting for ground, 5v, and signal using standard servo connectors.

Here is the 'Red board' and the mods – one track cut, a wire strap and a PPM diode or resistor.
We've standardised on PPM on D13 with this board:

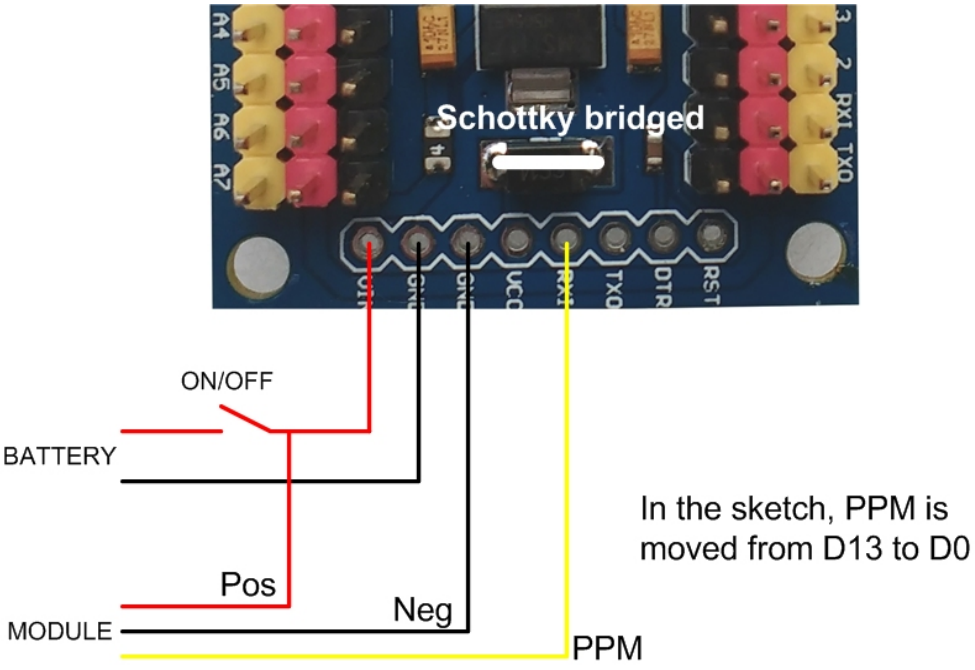




Here is the original, smaller ebay '328P' board made by Deek Robot:



The Deek-Robot 328P board has a diode which needs bridging, and has no convenient battery feed for the module, which has to be powered directly from switched pos (say at the on/off switch contact). D0 (RXI) is convenient for PPM with this board:

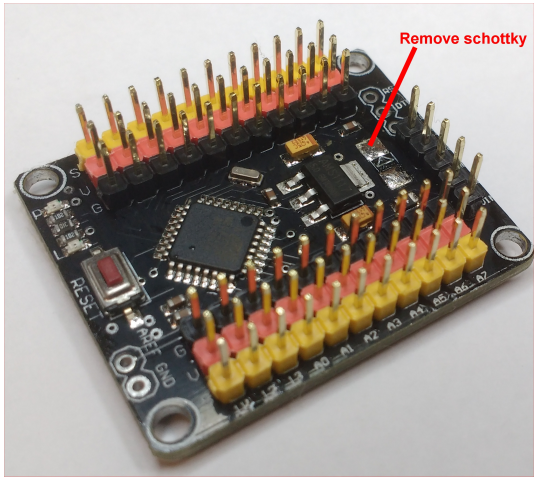


Further discussion on the various board options are on the Single-Channel forum, this is one example:
http://singlechannellersreunited.co.uk/phpbb3/viewtopic.php?f=9&t=625

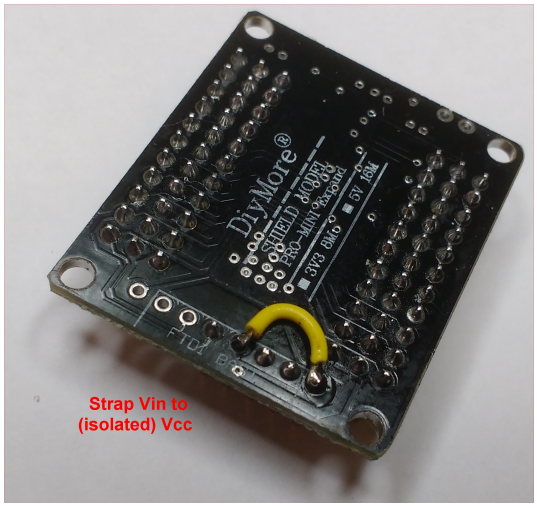
The DiyMore 328P board:

This one has proven to be the best option for encoder projects as it allows the three encoder connections to be made directly to the end header using a servo plug. The DiyMore is the one with the tiny TPH hole near Vcc (+5v) on the end header.

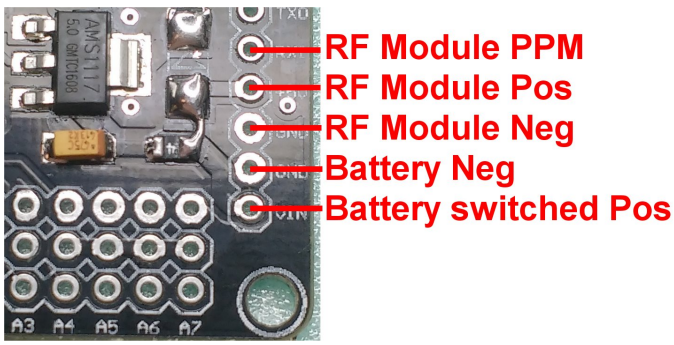
It needs a couple of simple mods, first the schottky diode is removed. This is much easier before the headers are fitted:



Removing the diode isolates the 5v pin on the end header so that we can use it for the battery supply to the RF module. There are two ways to do this, either using a short wire bridge like this:



… or by soldering a short wire bridge from the old schottky pad to the 4 ohm resistor. This is a bit more fiddly:

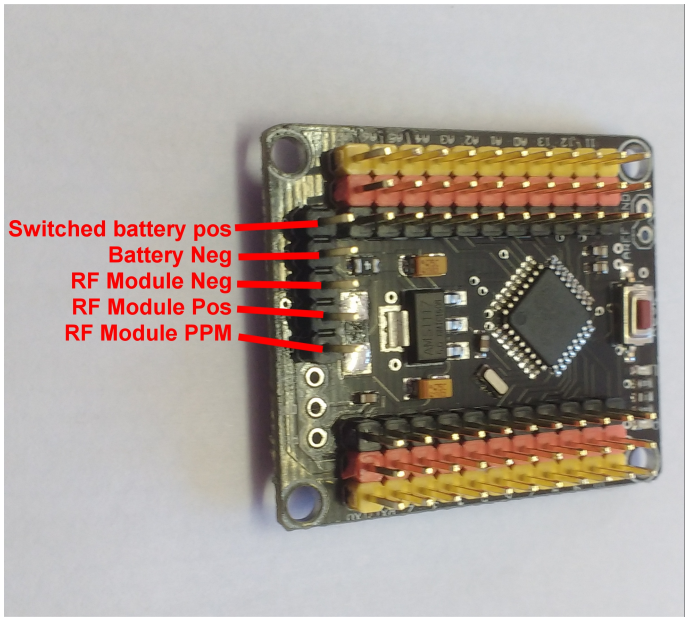


The DiyMore 328P board (continued)

The RF module connections for the DiyMore board are as follows, of the 5 pins, two are for battery switched positive and negative, and three are pos, neg and PPM for the RF module. PPM is on the pin labelled RX1 which is actually D0.

The remaining 3 holes are unused.

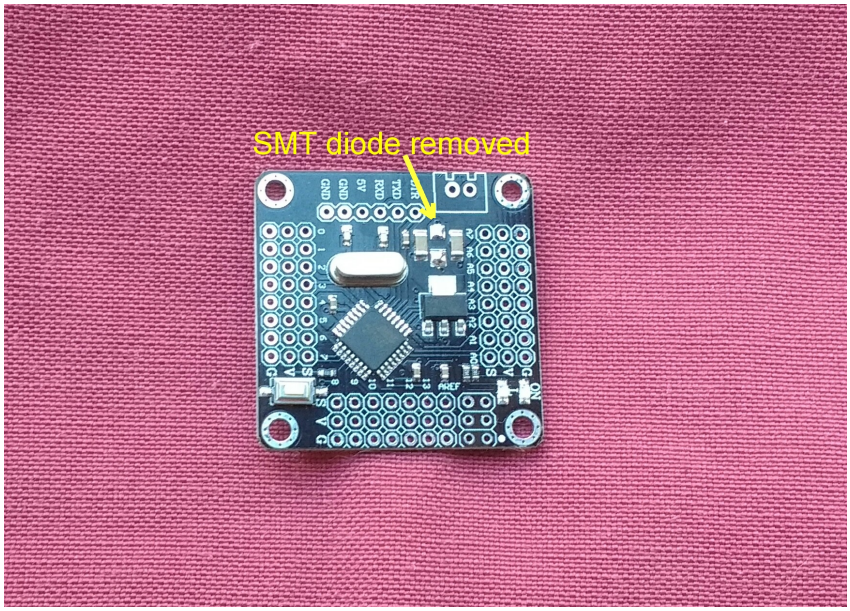
This is where the DiyMore is much neater than the Deek-Robot board which needs a separate pos supply for the module. A servo lead can be fitted to the module so it plugs straight into the encoder, really neat & flexible:



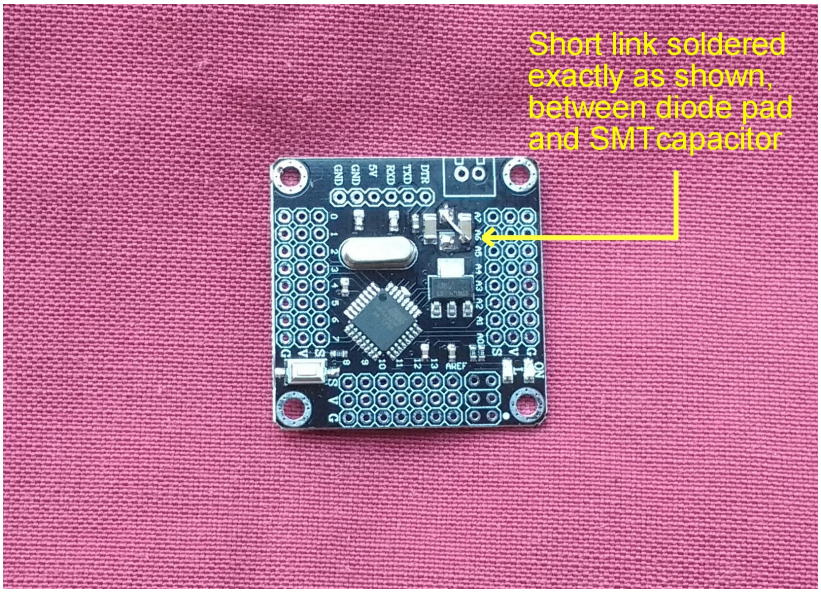
Note that in common with the Deek-Robot 328P board, A4 and A5 are interchanged on the board silk-screen annotation. Be careful not to confuse these two 328P boards, the Deek-Robot and DiyMore versions look similar but are routed completely differently. Use the appropriate mods for your board type.

A new type of board has appeared on ebay for 2018 and has often been supplied in lieu of the previous DIY-More board, this one is easily distinguished by having the 'servo headers' on three sides, and its 'Pro-Mini-Strong' label. Functionally its identical to previous boards, with some advantages – it has a proper 16mhz crystal rather than a resonator, so timing should be more accurate, and it has a separate power in connector. It a;lso has a 6-pin ICSP port. It needs a small modification to provide PPM, battery and neg for the RF module, which is most easily done before populating the board with the headers.

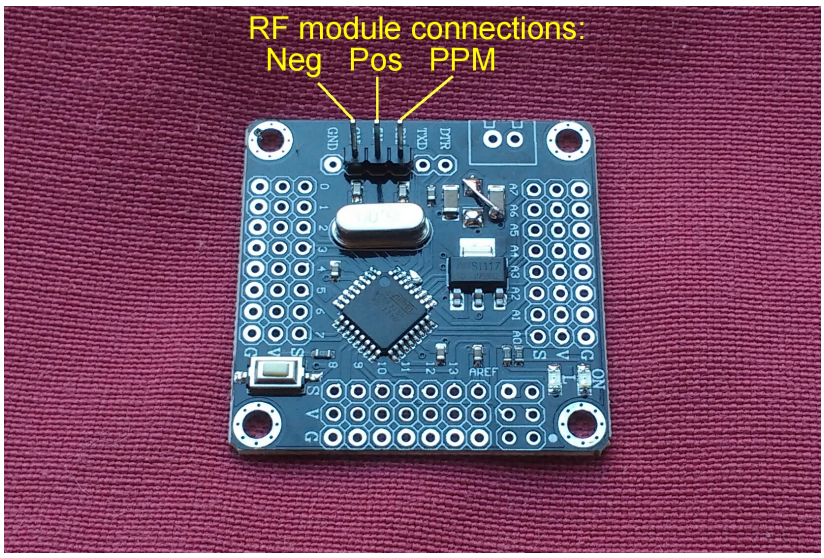
The mod is to remove an SMT schottky diode, then to bridge one of the vacated schottky pads to a nearby SMT capacitor:



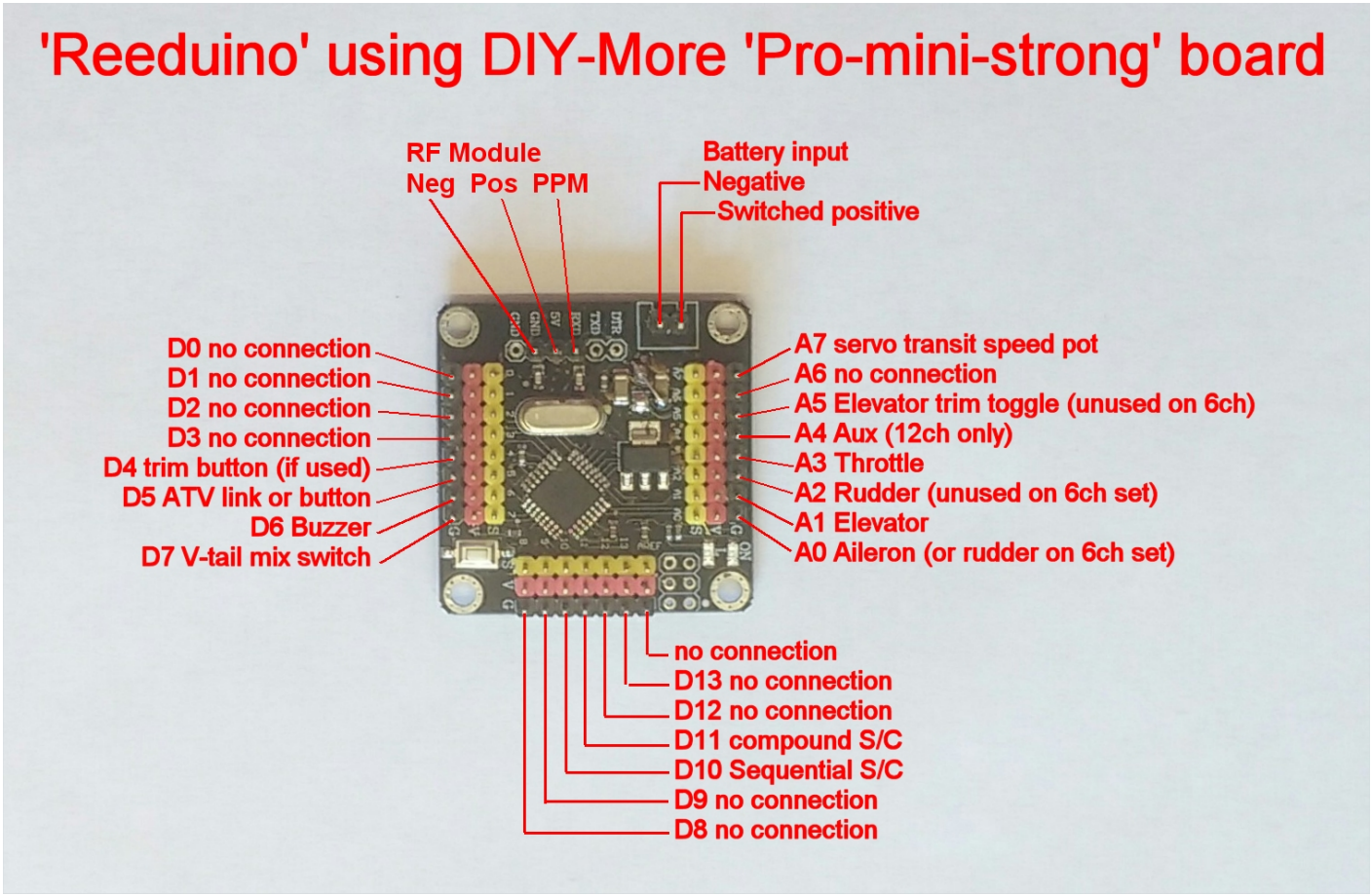
Having isolated the 5v input header, we can now use it to provide battery voltage to the RF module.
This is done with a short diagonal strap from the upper diode pad to the lower end of the SMT capacitor:



Heres a closer look:



And here is a summary of the connections. Please refer to the diagram and notes throughout the document.



Reeduino Facilities summary:

Up to 12 channel emulation. Used and unused toggles are configured in the 'activetoggles' array.
Optional full single-channel escapement emulation via a button, compound and sequential
Variable servo transit speed to match period servos such as the Bonner Duramite
Resolution is 1uS and is super smooth, even when the servo transit speed is set very slow.
Servo reversing by switching on with the required toggle thrown. Saved to flash.
Trims on every channel, by simultaneous use of elevator trim and the required channel. Saved to flash.
Trim-pips with longer neutral-pip
Elevator trim lock to make trimming channels other than elevator much easier
Optional alternative 'Tiny-6 style' trim button for Rudder/Elevator/Throttle sets with no trim toggle
V-tail mixer enabled by grounding D7 either with a Spektrum bind plug or via a switch
Adjustable Travel Volume (ATV) on every channel, range 20% to 112.5% travel. Default is 100%.
10 minute inactivity timer, sounds if no keys are thrown for 10 minutes. Any key resets the timer.
Range-check sweep mode, whereby the transmitter can be left whilst you walk away with the model
Master flash reset, reverts all reversing, ATV and trim settings back to neutral.
One character Morse identifiers are used to indicate the mode:
  D for reset to Defaults, R for Reversed, A for set ATV, V for ATV end, S for Scan, K for OK
Futaba AETR or JR/Spektrum TAER channel order
If the set is held in 'reset' for 10 seconds then a full ID is sounded

Channel Assignments

The configuration section highlighted at the beginning of the sketch has a statement:
static int Futaba = 1; // Set to 1 for Futaba=AETRXA. Set to zero for JR=TAERXA

If this is set to 1 as above, then the channel order is as follows:
  Channel 1 = Aileron (or rudder on a rudder/elevator/throttle model). This channel carries the S/C mix.
  Channel 2 = Elevator (and S/C kick-up)
  Channel 3 = Throttle
  Channel 4 = Rudder
  Channel 5 = Auxiliary
  Channel 6 = Aileron (repeated for dual aileron servos)

If this is set to 0 , then the channel order is as follows:
  Channel 1 = Throttle
  Channel 2 = Aileron (or rudder on a rudder/elevator/throttle model). This channel carries the S/C mix.
  Channel 3 = Elevator (and S/C kick-up)
  Channel 4 = Rudder
  Channel 5 = Auxiliary
  Channel 6 = Aileron (repeated for dual aileron servos)

The development is discussed in this thread:
http://singlechannellersreunited.co.uk/phpbb3/viewtopic.php?f=7&t=771
Please post your builds on there!

Thanks to Frank (Tiptipflyer) for all the test flying, and to the lads on the S/C forum for encouragement.

Cheers
Phil_G

Last update: 20/03/2017