Converting TowerPro 25A type 2 ESCs from PWM to TWI/I2C control

Introduction

Multirotor platforms such as the Mikrokopter (http://www.mikrokopter.de/), Big Quaddro (http://www.tt-tronix.de/), or UAVP (http://www.uavp.de/) require a very high rate of updating motor throttle settings. Most commercial ESCs are not capable of accepting PPM (PWM) pulses at a rate of more than about 150/sec. Multicopters are more stable if rates of approx. 300/sec can be ensured. This has led to the design of several ESCs which use the Two Wire Interface (TWI or I2C) between the microcontroler based avionics of these systems and the speed controlers. Best known example are the Holger speed controlers (see http://www.mikrokopter.de/). Originally these were limited to 5A continous, 10A peak power. Current versions can handle 10A continuous and 20A peak power. Their price is vey reasonable. However, some users have found the manual assembly with SMD parts difficult.
So far only YGE (http://www.yge.de/) offers commercial ESCs which have a TWI/I2C interface at power outputs up to 30A (YGE30i ESC). These can be integrated into the UAVP system, but use a slightly different software interface than the Mikrokopter (e.g. YGE throttle values 0..120, MK throttle values 0..255), and thus require changes to the MK software before they can be used.
Another approach to developiong TWI/I2C capable and affordable ESCs has been pioneered by Quax (Bernhard Konze) and several others, based on conversion of low priced commercial ESCs from PWM to TWI/I2C control. One of the most popular conversions has been the Arkai or TowerPro 17A ESC. TowerPro also offer a 25A ESC which is very similar to the 17A. Here a pictorial guide to converting a TowerPro 25A ESC type 2 (marketed since approx. Sept 2007) from normal PWM control to control via the TWI/I2C interface.

From the side with the sticker there is no difference between the TowerPro 25A type 1 or 2 ESCs.

Note the spead of bulges on this 25A type 2. On the type 1 all bulges are concentrated to the left of the W sticker near the capacitator.

TowerPro 17A and 25A ESC types, conversion guides, and firmware files

Note that this guide is for the newer type 2 ESCs which came to market from approx. September 2007 onwards. The older type 1 version had a slightly different layout, but in other respect were similar. Externally, in their yellow heatshrink, they are identical unless your recognize the different pattern of bulges due to the different location of voltage regulators on the processor side.
A conversion guide for the TowerPro 17A ESCs, which have an identical layout and for which the same conversion applies can be found here: http://home.versanet.de/~b-konze/low_cost/18a_regler.htm (in German but with pics, so probably reasonably understandable, certainly if held next to this guide).
http://home.versanet.de/~b-konze/low_cost/18a_regler.htm is also the location were the firmware files needed for the conversion of both the type 1 17A and 25A, and the type 2 25A ESCs can be found. At this time the most current version of the ZIP file with source code and the compiled HEX files is: http://home.versanet.de/~b-konze/low_cost/17a410_i2c_r07.zip.

Please note that there are some other ZIP files there for several other types of ESCs. You need the correct file. Using an incorrect file may damage the ESC or motor !!!

So again for the TowerPro 25A type 1 and type 2 ESCs you need a file 17a410_i2c_r##.zip (## denoting the version number) for controling them over the TWI/I2C interface. Other interesting files could be the 17a410_ppm_r##.zip, which allows to resinstal PWM based control.

Tools and materials needed for the conversion

To perform the conversion of these ESCs you need the following:

- TowerPro 25A type 2 ESC (multiple sources and different brand names; my source: http://www.unitedhobbies.com/ in Hong Kong).
- Small sharp siccors
- Very thin, isolated wire. I still had some old wire wrapping wire lying around which is ideal for this purpose. The wire with plastic coating should not be much wider than a microprocessor leg, and the bare wire should not be wider than a processor leg.
- Thin tipped soldering iron (doesn't have to bey anything fancy. I use a very simple Antex M12 12Watt iron without any controls).
- Wet sponge with the soldering stand.
- Thin soldering tin intended for use in microelectonics.
- Desoldering wick (just in case you e.g. bridge some processor legs).
- Pointed tweezers (preferably the type which close when you release pressure.
- Microdrill with a small (e.g. 0.5mm) drill.
- Some foam double sided tape.
- Some clear solvent free varnish (e.g. one of those small bottles for hobby painting).
- Strong magnifying glasses. If you can find them and can work with them, stereo magnifying glasses could be seriously considered.
- Good lighting (one of those hobby lamps with large built in magnifying glasses is ideal).
- Clean workspace.
- Printout of critical pages of the manual (always easier to check things off on paper than on a screen, or to even have a critical pic under your work to make sure you do what you see).
- Simple small volt/multi-meter. Mine is very small, and cost less than 15 Euros. You do need small thin tipped measuring leads to be able to accurately measure resistance or voltage between different processor legs.
- SerCon serial converter (http://www.mikrokopter.de/) or other ISP programming interface and computer with which you can reprogram the ESC.
- Software for programming ATMega8 chips, e.g. the free PonyProg (http://www.lancos.com/prog.html).
- ZIP file with firmware and settings for the ATMega8 fuse bits.
- Patience !!! Do not hurry. Converting an ESC is microsurgery. Hurry and the "patient' may die. And mistakes can cost you more than just an ESC. I-ve burned a motor or two due to mistakes.

Tips for soldering wire to SMD mounted parts and processor pins

To solder wire to the processor pins the technique I-ve found to work best is to carfully coat the target processor pin and the bare end of the wire with a thin layer of solder.

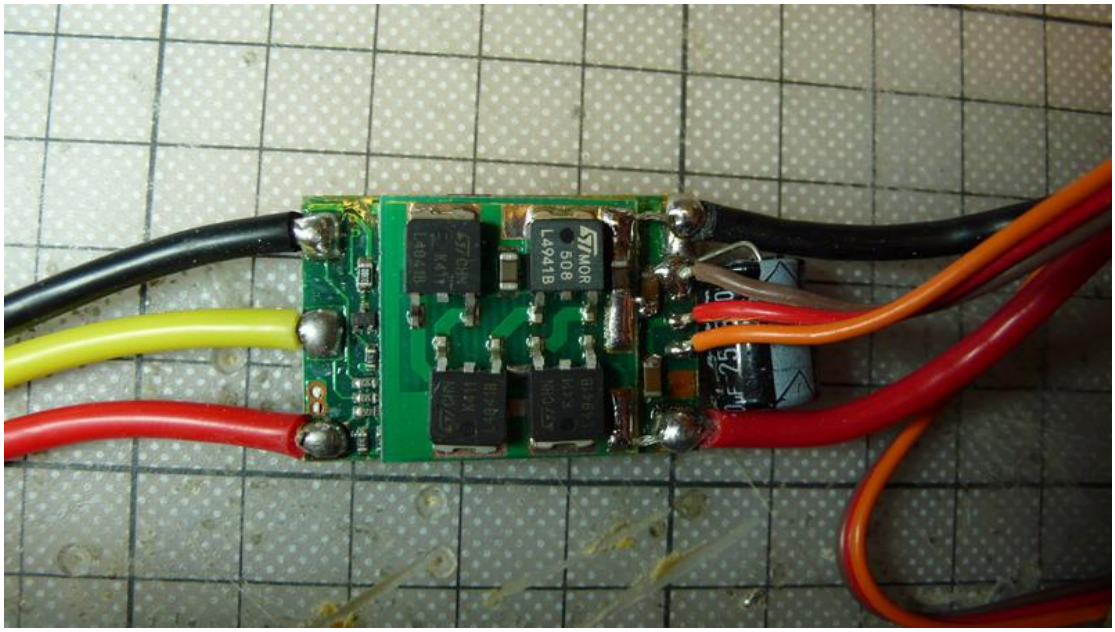
First clean the tip of the solder iron on your wet sponge. Then just nick it to the solder so that there is a VERY SMALL drop of solder on the point (should be hardly visible, otherwise it is too much). Carefully touch the correct processor pin with the tip of the soldering iron moving up and down along the leg to coat it. Take care not to bridge solder to neighbouring legs. Carefully inspect your work. If in doubt aboutsolder bridges, get out the desoldering wick, soak them up, and redo this step.

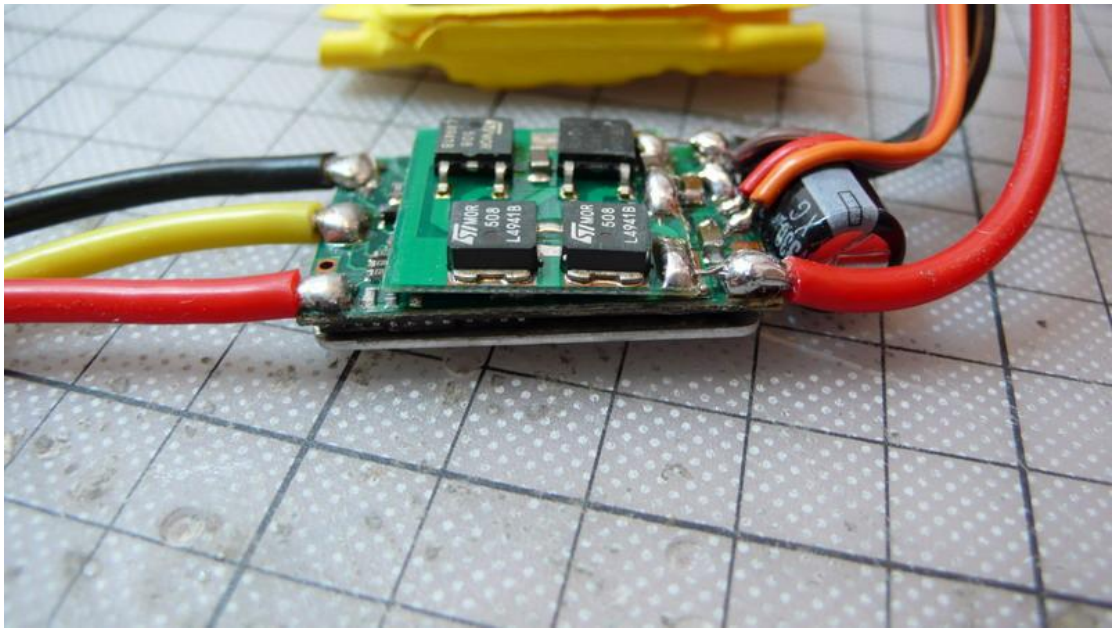
Now clear 1-2 mm of the end of the wire and give that a very thin coat of solder. Cut it back to 0.5 to 1mm bare wire and bend it in such a way that you can comfortably push it up against the leg so that most of the non-isolated part is flat against the leg. If necessary use tweezers to hold the wire, but don't use them too close to the end as the might bite into the heated plastic coating (keep at least ¾ of a cm away from the tip). While pushing the wire against the processor leg, just tip it with the soldering iron. That should melt the solder on both the wire and the leg and result in a good bond. Wait a little for the solder to cool down.
Inspect the bond under magnifying glasses. The bond should be over most of the length of the wire tip. If it;s just a small portion it will come loose at some point which might result in a serious failure.

The actual conversion of TowerPro 25A type 2 ESCs to TWI/I2C control

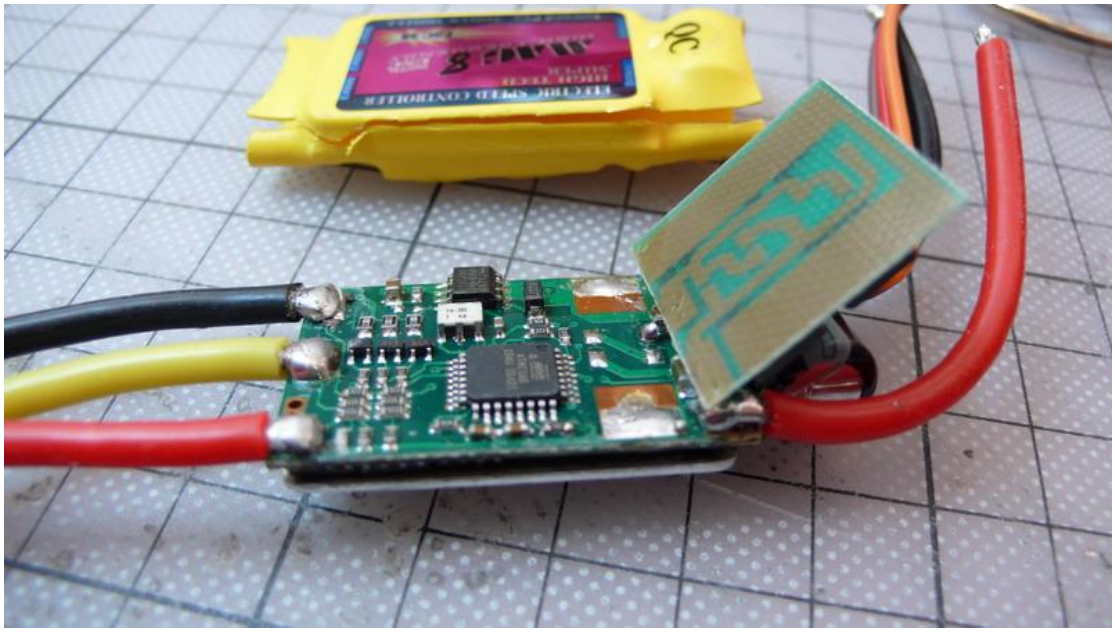
First cut away the heatshrink. Best to do this along the side or over the metal plate on the the side whith the sticker. That way there is no risk of damaging any parts.

Note that for our purposes the sticker on the heatschrink is actually on the back side of the ESC, the side with the metal plate heat sink.

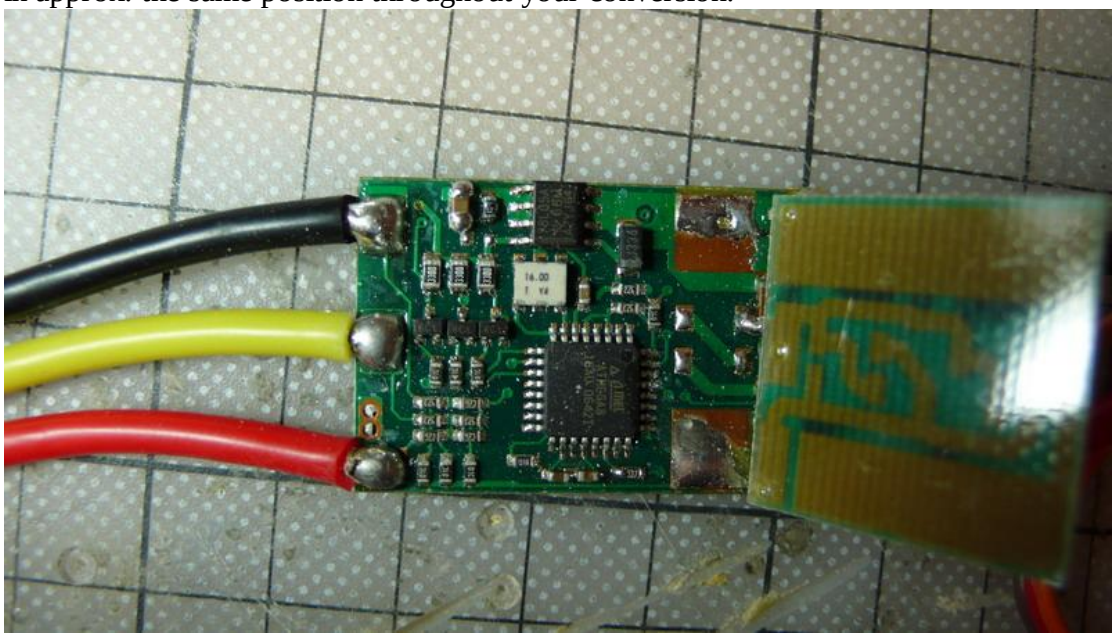




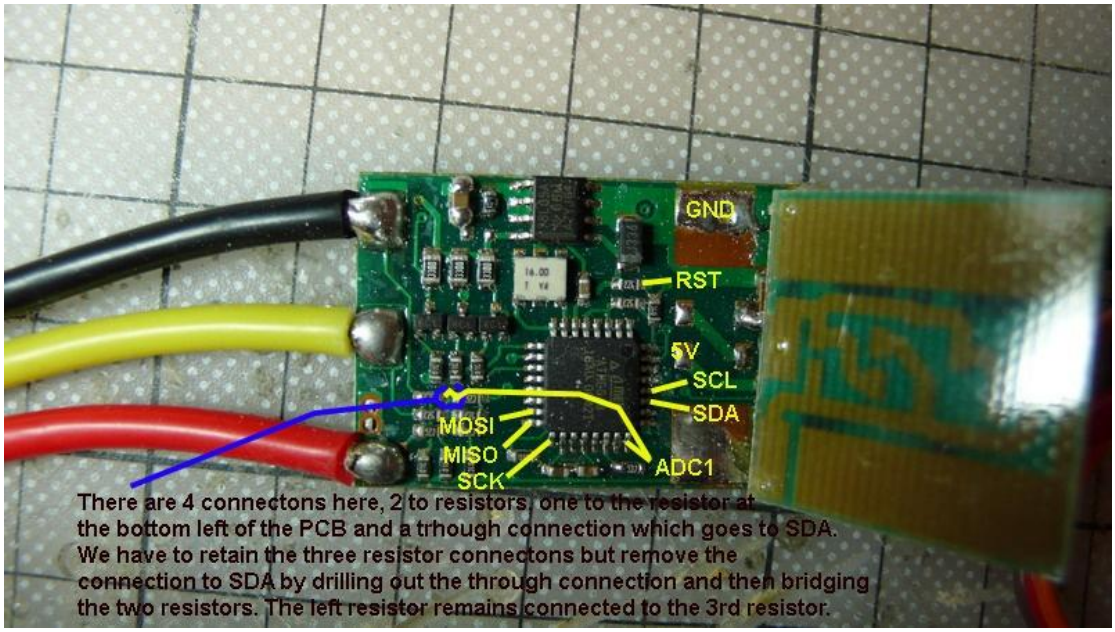
After removal of the heat shrink the topside of the type 2 ESC is clearly recongisable. this type has a second PCB with 4 voltage reguloators connected to the main PCB with 3 bare wires.



Carefully fold back the voltage regulator PCB. Avoid bending it back and forth too much as this will sever the three wires connecting it to the main PCB. Try and keep it in approx. the same position throughout your conversion.



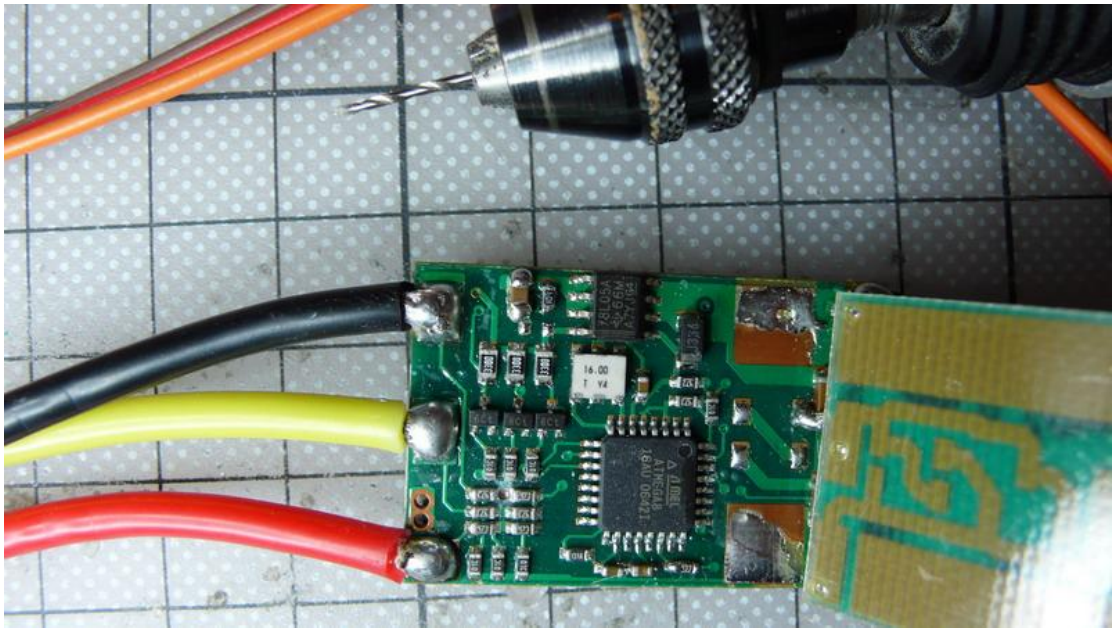
If you are certain you are not going to use the 5V BEC power supply from the ESCs, you can consider removing the voltage regulator PCB. Personally I find that a waste. It doesn't hurt to leave it in place. And it gives you the option of using the BECs. But do note that if you use the BECs for additional 5V power to a system which already has a 5V power suppply, you must put a sufficiently large Skotschky diode in between the BEC and the system's 5V rail.



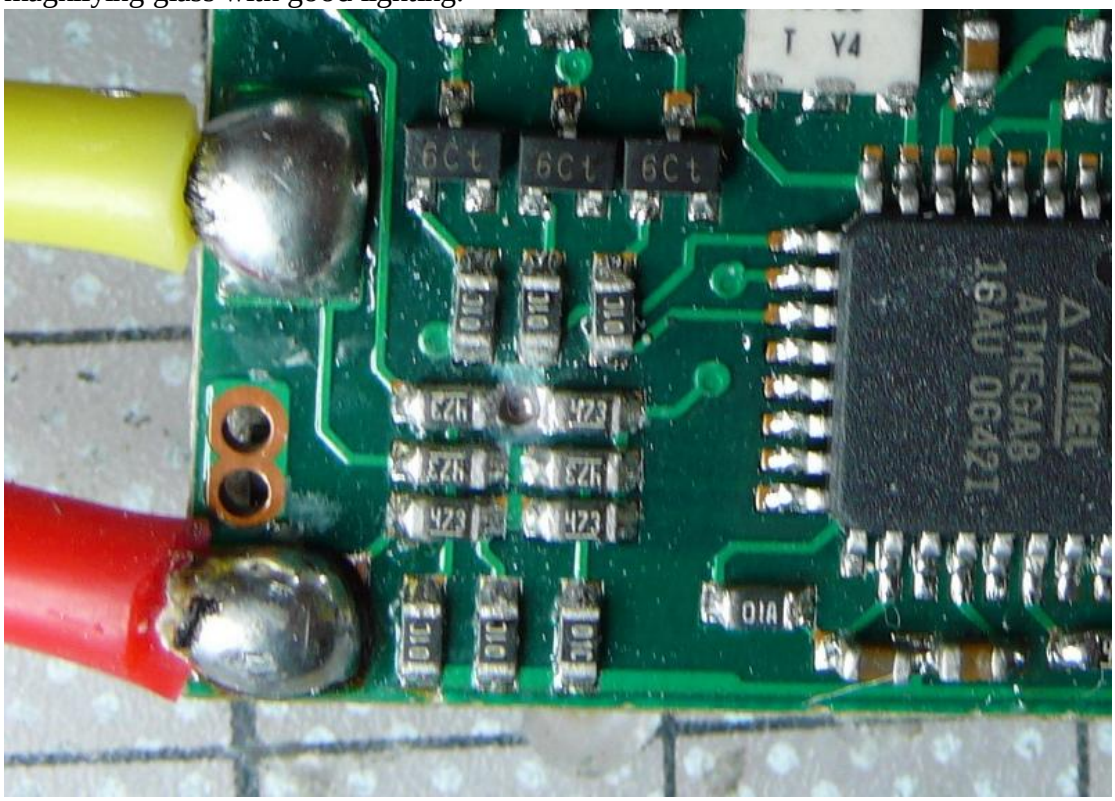
For the TWI / I2C interface we need at least the SDA and SCL lines and a common ground connection between the system and the ESC. As supplied the SDA pin is in use as an analog converter. So we have to disconnect the input from that pin and move it to the ADC1 pin. See more on this two paragraphs down.

For reprogramming the microcontroler we also need to bring out an ISP interface consisting of MISO, MOSI, SCK, RST, 5V and GND lines. For the RST lines we are picking that off of the end of a resistor instead of the pin next to the SCL pin. Soldering three wires next to each other on processor pins is just a bit too challenging if not needed.

The most difficult part of the conversion is to move an analog digital conversion port from the SDA pin to the ADC1 pin of the processor. To do this you must drill through a pad between two resistors and deep enough to completely disconnect a connection going through the PCB (but not drill completely through the PCB). See the little blue circle in the picture above which identifies the pad betwen the two resistors you have to drill through. You then have to put in a little bridge between the two resistors which have now been disconnected from each other, and connect that bridge with the ADC1 pin. This is going to be the first step in the converions process.

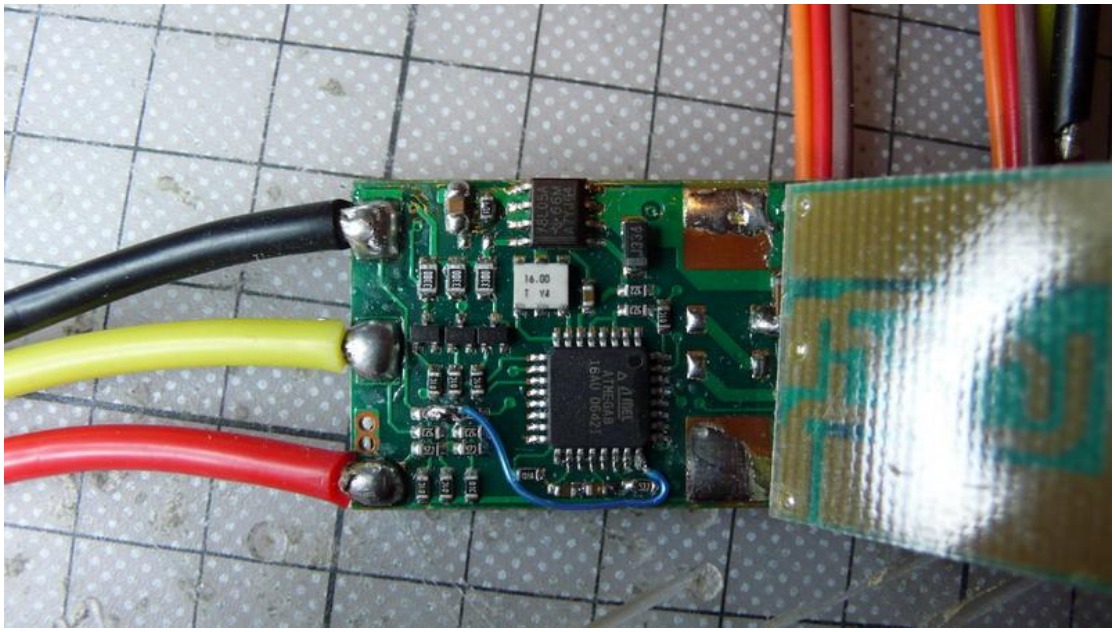


Take your time to drill away the pad and the through connection. You do NOT want to damage other pads or connections. Inspect your work carefully under a strong magnifying glass with good lighting.

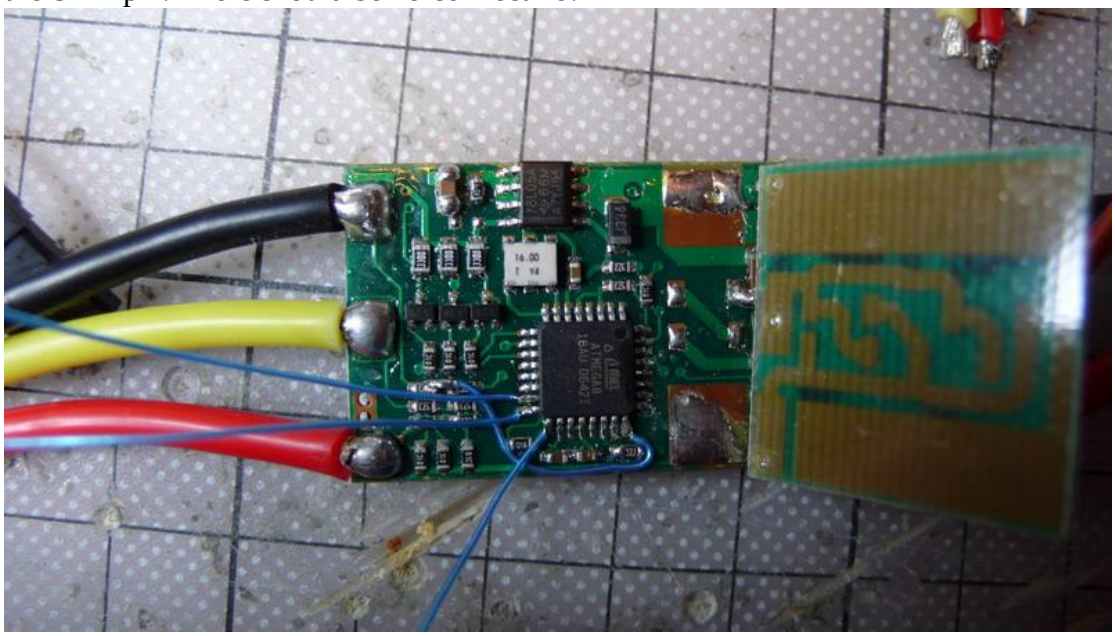


If you are sure the through connection has been completely disconnected from the pad between the resistors, clean the ESC with compressed air / a dry brush.
Stop and do a more scientific check now.
With a voltmeter check the resistance between the two resistorends adjacent to the hole and the top end of resistor at the bottom left in the pics. The left of the two should show 0 Ohm, and the right should show no connection.
Also check the resistance between these resistor ends and the SDA pin on the processor. All should show no connection.

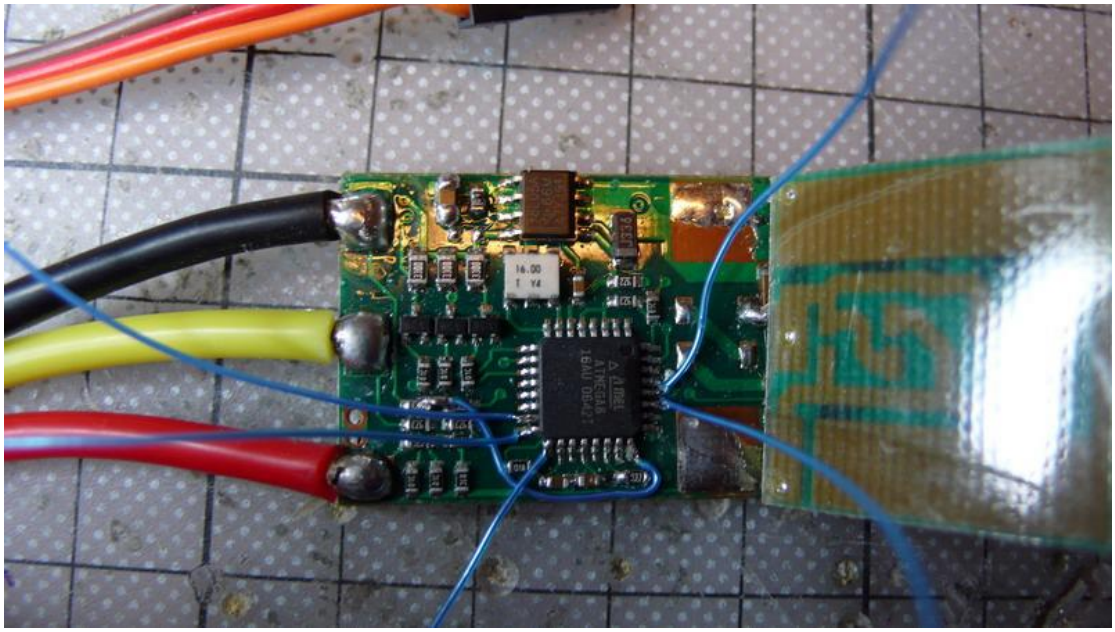
Once this checks out OK put a little drop of some clear varnish into the hole to make sure that no accidental reconnection can occur. Make sure you don't put varnish on the resistor ends.



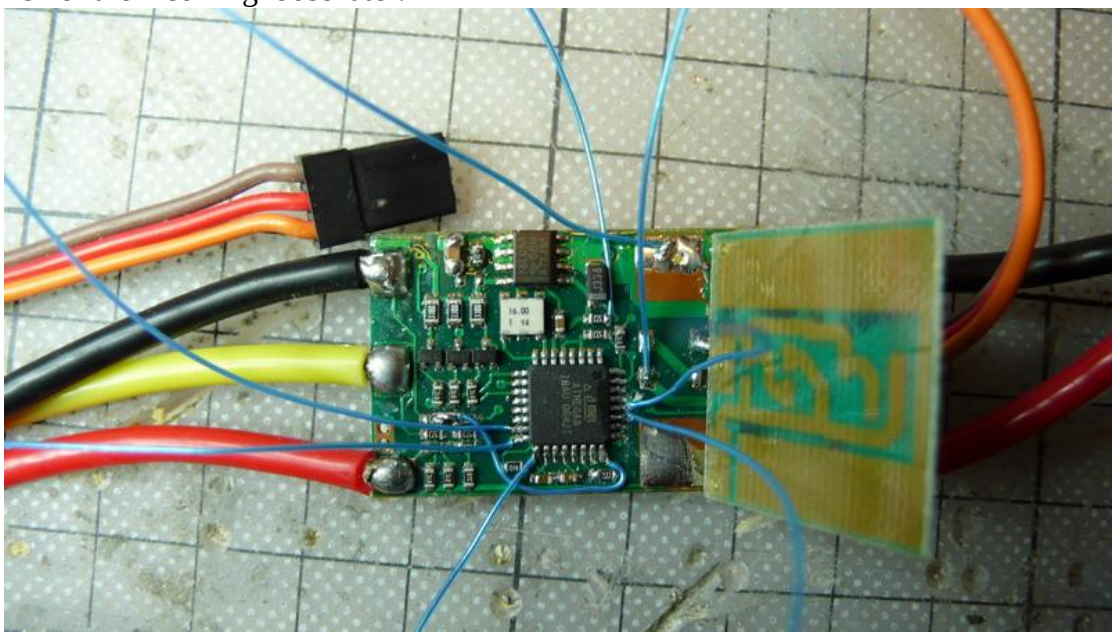
Once the varnish is dry you can solder some very thin wire over the two resisotor ends to create the jumper bridge, and then very carefully onto the ADC1 pin of the processor. Carefuly insect these connections for unwanted solder bridges to neighbouurng resistors or pins. If necessory bring out the desoldering wick, such up the excess and redo.
Also redo the measurement of resistance between the solder bridge / ADC1 pin and the SDA pin. There should be no connectino.



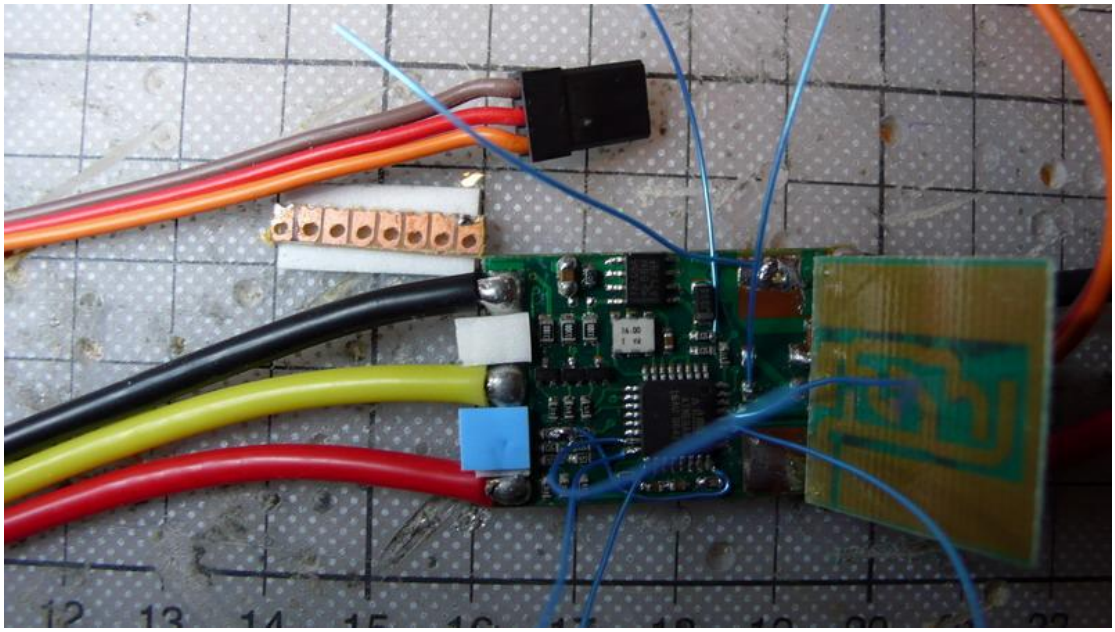
Solder the MOSI, MISO and SCK lines to the respective pins. These are quite simple connections. Make sure the wires do not push down on the already installed wires wihile soldering as this may meld the insulation and result in later short circuits.



Now connect the SDA and SCL lines to the microcontroller. They are a bit more difficult due to the voltage regulator PCB. I have found that bending the end of the wire a bit, and having SCL go to the top, and SDA to the bottom is easiest. Carefully inspect all connections to the processor pins under the magnifying glasses. If you have any doubt about soldering bridges, bring out the desoldering wick, suck up, and redo. Also make sure the bonds have some length so that there is little or no risk of them coming loose later.

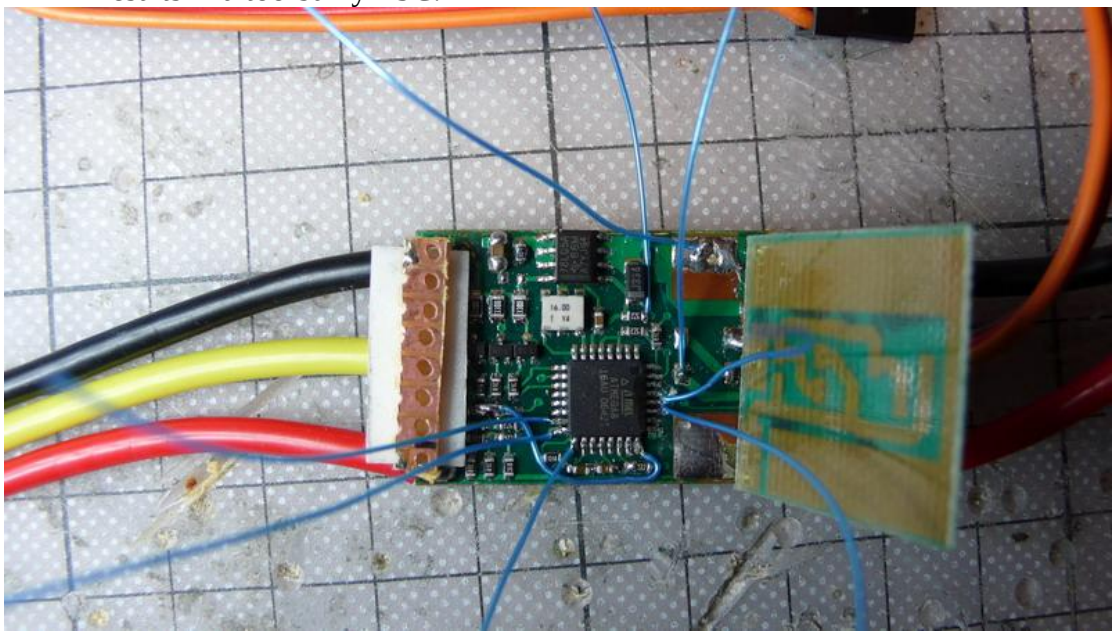


Connect the wires to the identified pads for 5V and GND, and to the end of the resistors where we are picking up the connection to the RST pin. Very simple. Consider a little bit of varnish over the 5V pad to further ensure proper insulation from the SDA and SCL lines.



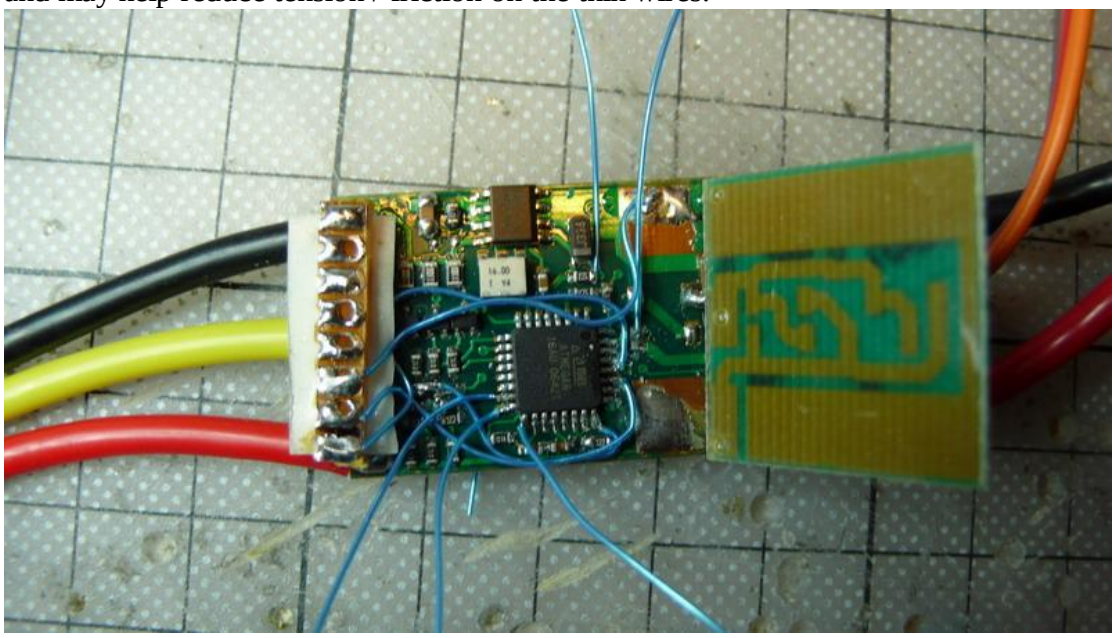
You now have three options:

a) Connect the wires needed for programming to your programming connnector, reprogram the processor, and remove these wires retaing only the wires needed for the TWI/I2C interface and their connectoin to a connector cable of your choice. This does mean less “spaghetti” retained in the ESC. However if new firmware comes out you will have to redo part of the microsurgery.
b) Connect the programming wires and the TWI/I2C wires to a scrap of prototype board to which you permanently connect the TWI.I2C connector cable and temporarilly connect the ISP connector. This is my prefered approach. It adds little weight, reduces tension on the micro-wires, and does not require all the microsurgery to be redone for upgrading firmware.
c) Same as b) but with a permanent ISP connector included. This might be worthwile when you are yourself actively developing firmware but otherwise results in a too bulky ESC.



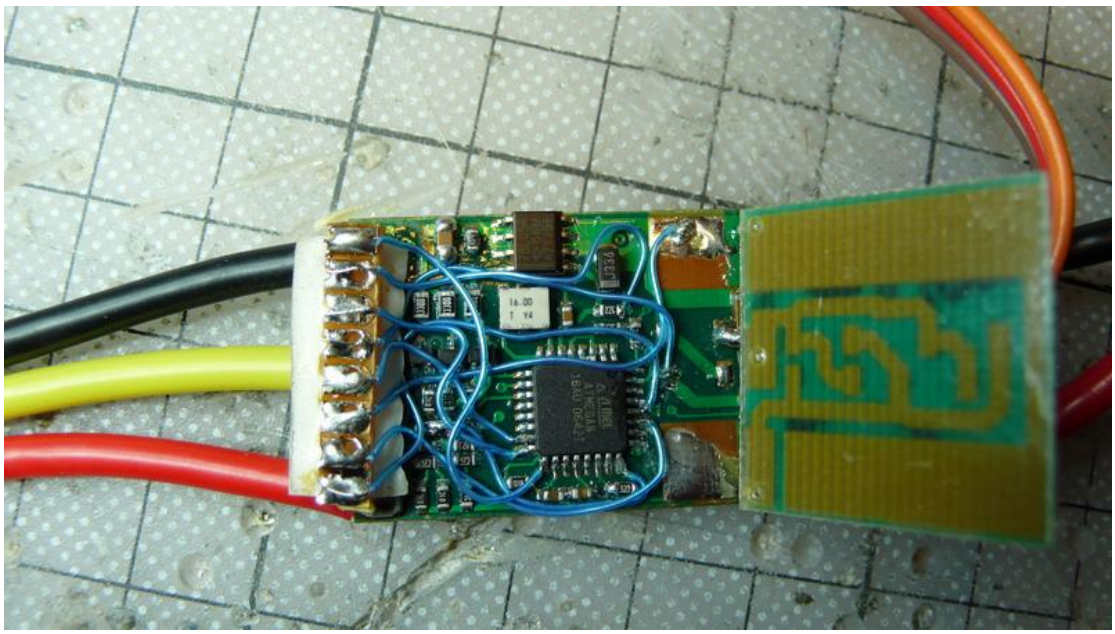
I use foam double sided tape to keep my small scrap of prototype board with 8 pads in place. As this sits over the motor-power lines you first have to put two double height

pieces in between the wires, and then a full length piece on the bottom of the padboard. I purposefully have kept that wider than the padboard as it helps ensure the board is kept well in place, ensures it is properly isolted from the motor power wires, and may help reduce tension / friction on the thin wires.

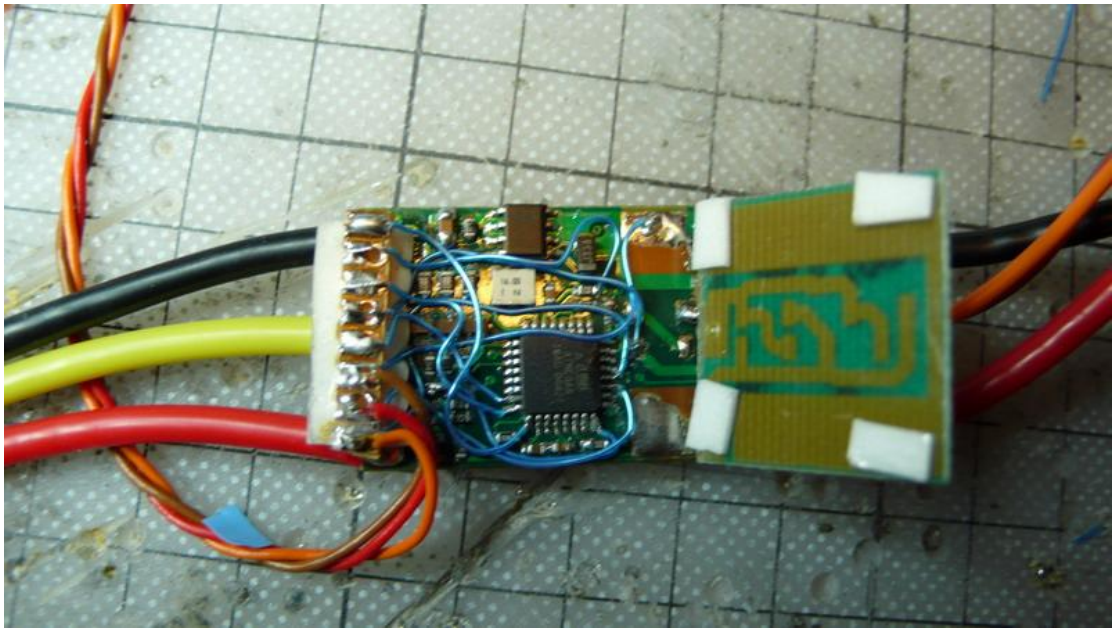


From top to bottom the pads are: SCK, RST, MISO, 5V, MOSI, GND, SDA, and SCL.

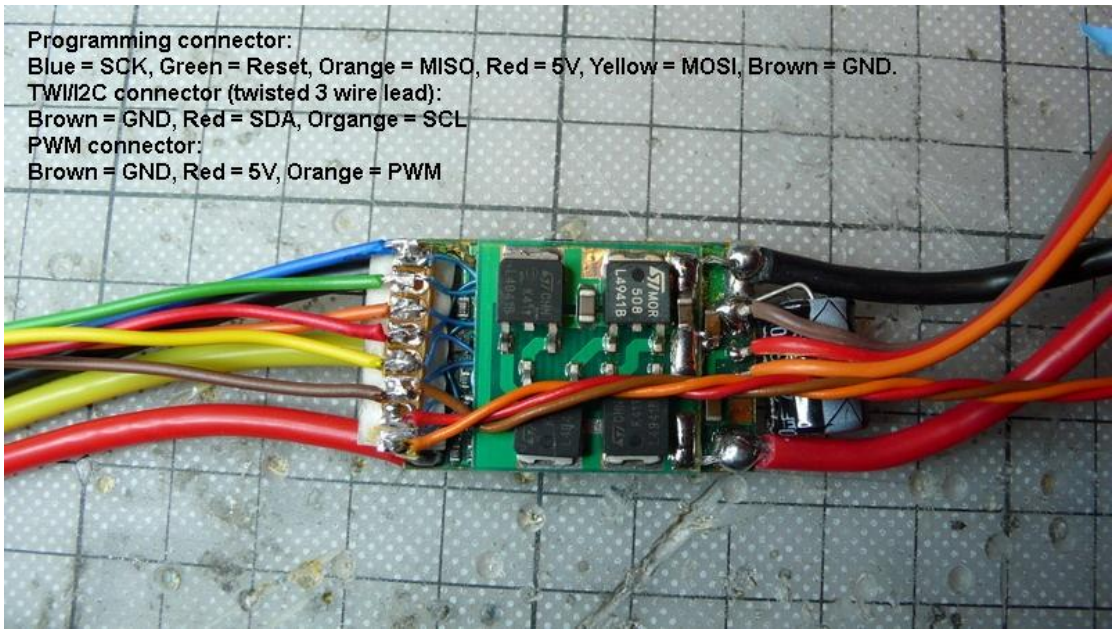
Carefully shorten the thin wires to such a length that the can be routed easilly between the larger parts on the PCB and brought up to the level of the pad board without stress.



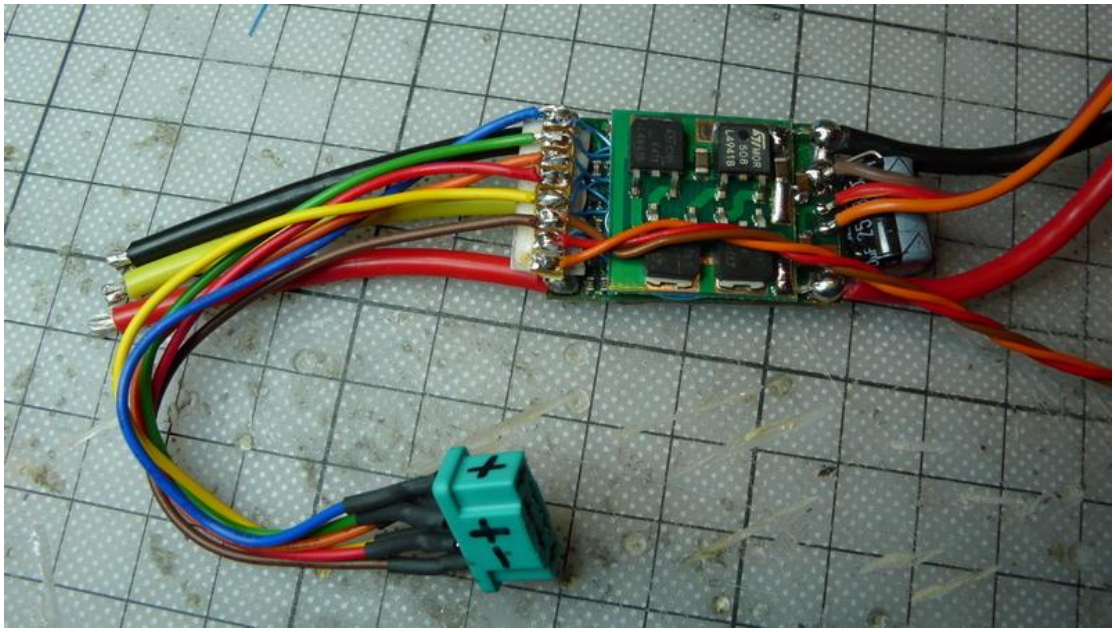
Spaghetti, or blue worms, whichever is your fancy. But hey, if it works, why not use it ;=)) If possible do not run the wires parallel to each other too much. Better to have more them cross at angles. Check the pads for solder bridges.



Connect the TWI/I2C cable. I use thin twisted servo wire: Brown = GND, Red = SDA, Orange = SCL. At the servo connector end the same sequence is maintained.
Note the small scraps of doublesided foam tape on the bottom of the voltage regulator PCB. This may help reduce the risk of friction eventually going trough the covering of the thin wires.



Fold the voltage regulator board back down taking care to lead the TWI / I2C connector cable over the voltage regulator board. The foled back board should not be touching any of the thin wires with its edge.
In this image I-ve also already connected my programming connector. See the color code description in the picture for the different connector cables.

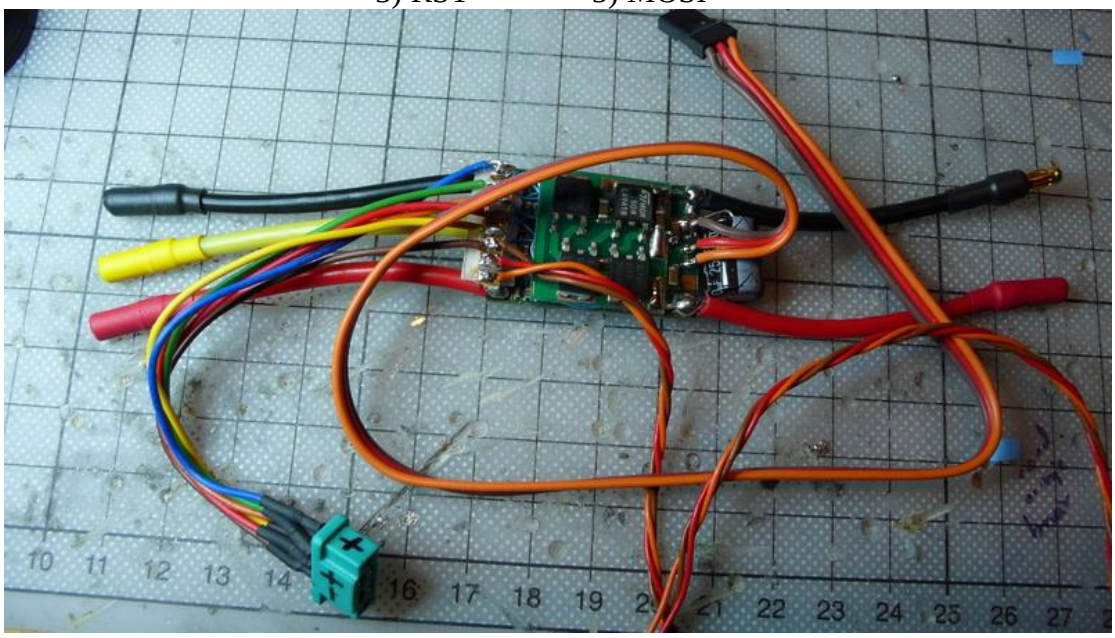


I-ve found a Multiplex battery connector to be the ideal ISP connector: 6 pins, and no risk of incorrect connection. As you can see from the colours, the layout (seen from the rear of the connector is:

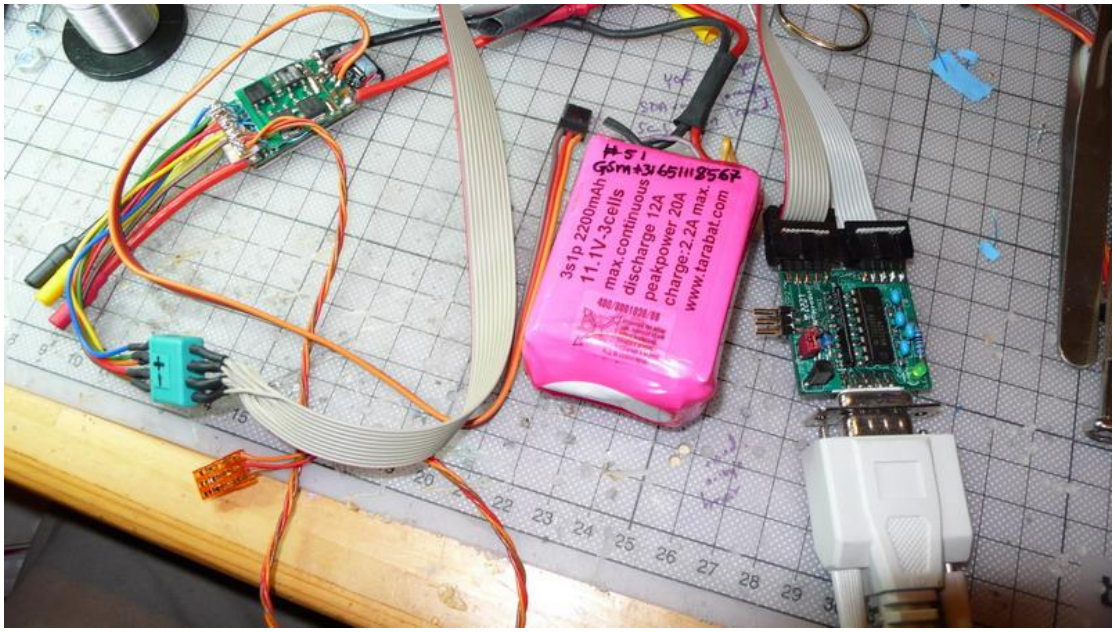
2) RST 4) 5V

1) SCK 6) GND

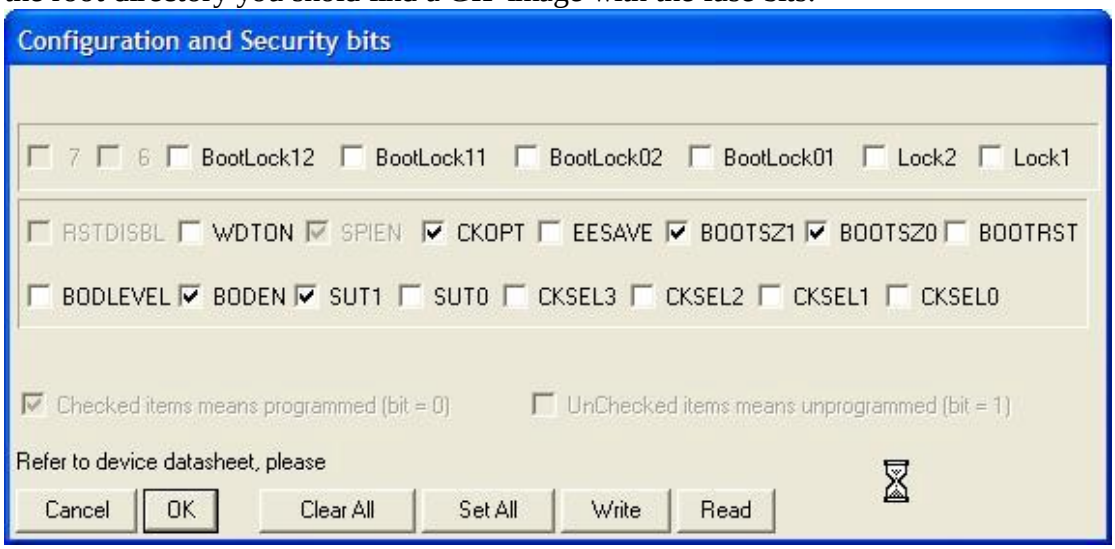
3) RST 5) MOSI



Note also that if you haven't done so yet, you now have to connect the motor and battery connectors to their respective wires, and put heatshrink around them. I use 3mm banana plugs for both the motor conectors and for the power connectors. They should be able to handle up to about 35A which exceeds the specs of the ESC. For heavier ESCs I prefer 4mm banana plugs for the power connectors.



Oops... missed a step here: Do NOT connect all sorts of stuff to the ESC. Consider using a limited power stabilised 10-12V power supply and not a precious lipo. Also consider some sort of a fuse in the powerline, such as a car lightbulb which should burn through if the power draw is too high.
First connect only the power source and check that you indeed have 5V between the 5V and GND pad. If that is the case, then you can connect your programming interface. In this case I-m using a SerCon (serial converter) from the Mikrokopter site (http://www.mikrokopter.de/). Note that the red jumper on the Sercon is set to programming mode and eht green LED is on indicating that the SerCon is getting its 5V power over the ISP interface.
Start up your programming software, e.g. the free PonyProg program (http://www.lancos.com/prog.html). I-m not going to go into great depth about the interface of that. It has it's own tooltips and helpfiles.
Now go to the firmware ZIP file (e.g. http://home.versanet.de/~b-konze/low_cost/17a410_i2c_r07.zip) and unpack that if you haven't done so yet. In the root directory you shold find a GIF image with the fuse bits.



Set the bits as shown, and write them to the ESC. If this works, then you know you've probably got a success on your hands ;=))

Now op the HEX file you need. Below the root of the unpacked ZIP file you will find a subdirectory “hex”. All four files in this subdirectory have filenames like bl-17a_r##_p40_m#.hex, where ## denotes the firmware version number, and the last # denotes the motor number. Make sure you open the right file for the motor number you intend to use the ESC for.
On the Mikrokopter, seen from the top, the motors are numbered:

1) Front

4) Left 3) Right

2) Rear

Write the HEX file to the ESC. This normally is followed by verification. Due to a bug in PonyProg and/or the SerCon interface I always get an error message on the verifcation. There seems to be a change needed to some INI files which corrects this, but I-m a bit lazy....
Disconnect the ISP plug and power down the ESC.

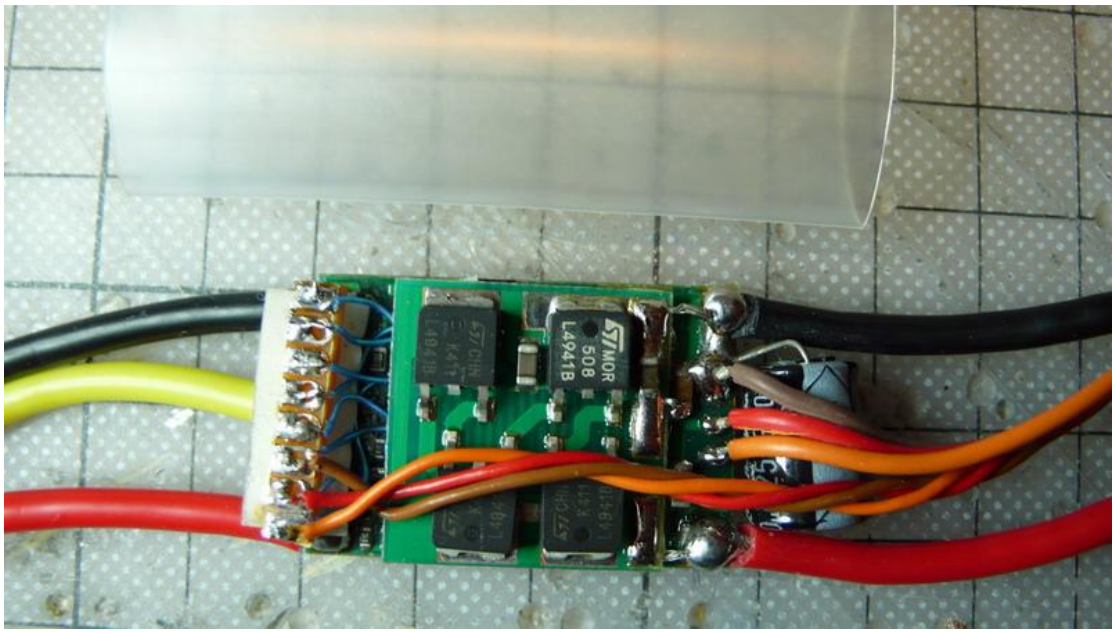
Testing the converted ESC (e.g. with the Mikrokopter and MKTool)

This section is rather Mikrokopter specific but is hopefully general enough for others to also be able to follow it as a guide.

Make sure the MK controler / motor / frame are securly fastend to e.g. a teststand. You don’t want things sudding going airborne. Make sure the props of any of the connected motors cannot hit anything.

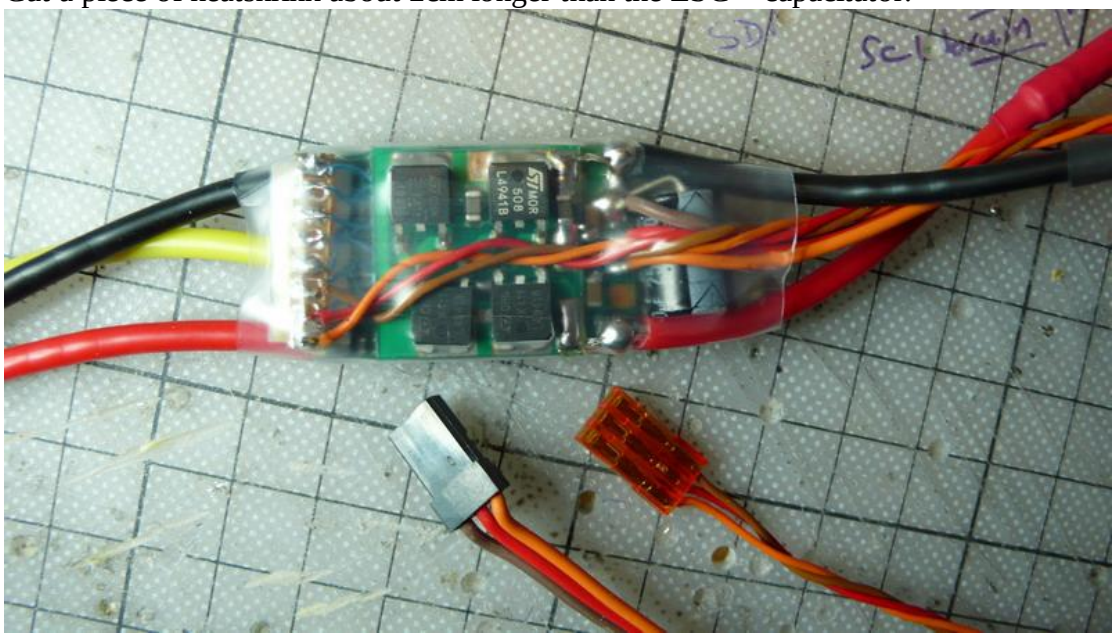
Connect the MK to the computer via the serial interface and startup the mikrokopter setup tool.
Connect the TWI/I2C cable to the mikrokopter controler, power lines to the 11V power rail, and motor connnectors to a test motor. And yes, consider using a limited power source, or a relatively low amp fuse in the power lines when testing. One time I didn’t and literally burned a motor due to an error I had made in the conversion (lost the connection between those three resistors :=((On the bright side, it was really a test motor which I wasn’t to sure about using for real flight ;=))
On applying power to the power-rail the ESC should have powered up giving a little beep and prop wriggle if it programmed correctly. If not do another attempt at setting the fuse bits and reprogramming. I-ve found that usually programming is OK straight off, but sometimes I need 2 or rarely 3 attempts.
If the ESC beeped happilly, turn the Mikrokopter flight controler on and wait until it beeps and shows a red and a green light (connected to the SerCon so no receiver input). You should see sensor values becoming active onscreen. Go to the motor tab and click the motortest checkbox. Carefully increase the throttle slider for the motor number you are testing.
If all went well the motor should start turning regularly at a low throttle setting. Don’t move the throttle slider too quickly there is a delay between computer and MK. You don’t need sudden jumps in throttle.
Throttle down, disconnect everything, sit back and grin happilly: you have a successful conversion ! ;=))

Finalizing the conversion



Disconnect the programming interface wires. Make sure the thin wires remain correctly in place and no soldering bridges develop. Of course leave the TWI/I2C interface connector. If you are sure you don't need the BEC at this time you can consider removing the normal servo connecotr wires. I'm lazy, so I just leave them in place....

Cut a piece of heatshrink about 2cm longer than the ESC + capacitator.



Try and lead the TWI/I2C cabe between the voltage regulators before heatshrinking. Also be careful not to melt the servo wires during the heatshrinking. Well applied heatshrink will help keep the little pad baord firmly in place.

The final result. As a nice little touch I-ve pulled the original stickers off of the old heatshrink and stuck them back on the modified ESC, which is now a TowerPro 25A MAG8***i*** ESC for ***m1***. And it has passed the Quality Check, i.e. it has been shown to work correctly.

Final words

Of course QC could also as easilly been taken to read “Quax Conform”. Without the great work of Quax and several other quadrotor geniuses, we would not see the rapid development of these systems that we have seen in the last year or so.
Of course neither those who developed the firmware and figured out the changes needed to the PCB, nor the author of this manual can accept any liability for your errors or for failure of the ESC. And you probably voided your warranty by modifying it. However, if all went well you should hopefully enjoy many good and safely powered flights with this TWI/I2C enabled ESC.

Arthur P.
2007-11-03.