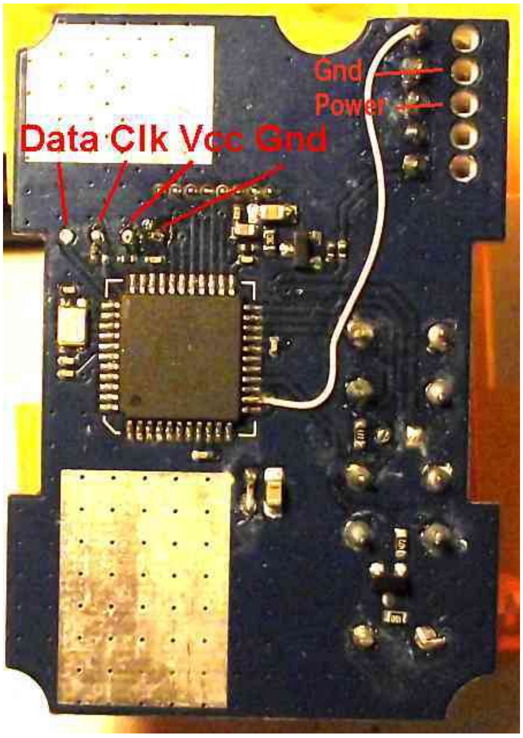
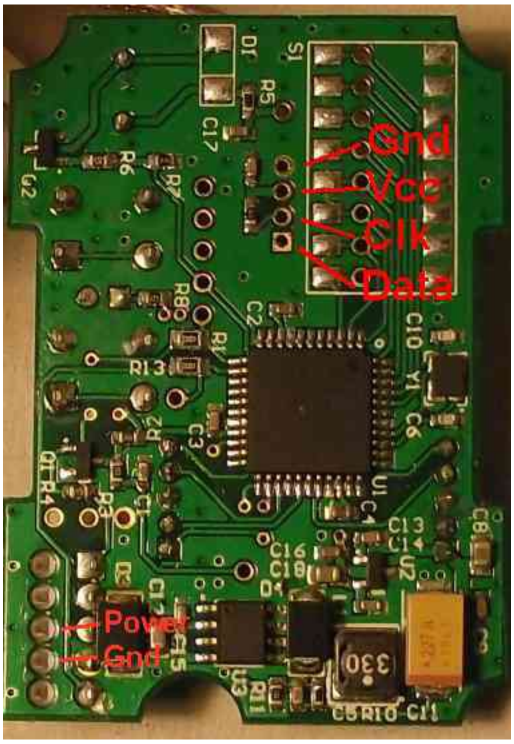
Details for updating an Orange DSM module to MULTI protocol.

Here are pictures of the two module types I have.
The blue one needs a wire added, and the programming pads need to be soldered.
The green one has the connection to the 5-pin module connector already present. The programming vias are on a 2mm pitch. I managed to bend the pins of a 0.1" header to fit so I don't need to solder to them.





The wire added to the blue module is for telemetry output.
As these modules use an ATXMEGA32 processor, it is not possible to use a standard USBASP programmer. If you have a programmer that handles the PDI programming method, then use that. For those without, I have managed to put the code into the ersky9x for the 9XR-PRO, the 9Xtreme board, the SKY board and the AR9X board. The 9XR-PRO and AR9X are able to connect directly, but the 9Xtreme and the SKY boards will need the addition of a transistor buffer/inverter to be able to drive the CLK signal properly.

I have also written a bootloader for this board that allows the firmware to be flashed from the radio without needing the wiring to the PDI connections. It is necessary, however, to use the PDI pins to flash the bootloader.
If you have an Arduino Pro Mini, there is a sketch for that which flashes the bootloader to the module. The sketch is in the file "pdiMini.zip", you will need to flash this to a Pro Mini.
To use, connect:
Ground from the module to ground on the Arduino.
Vcc (3.3V) from the module to Raw (or maybe VCC) on the Arduino (I tested using Raw).
PDI Clock from the module to IO2 on the Arduino.
PDI Data from the module to IO3 on the Arduino.

Power the module, the Arduino will be powered from the 3.3V from the module.
After a couple of seconds, the LED on the Arduino will flash, around once per second if the bootloader flashed OK, and 5 times per second if it failed.
In theory, the Arduino is running too fast (16KHz) for operation from 3.3V, but it should work, and only needs to flash the module once.

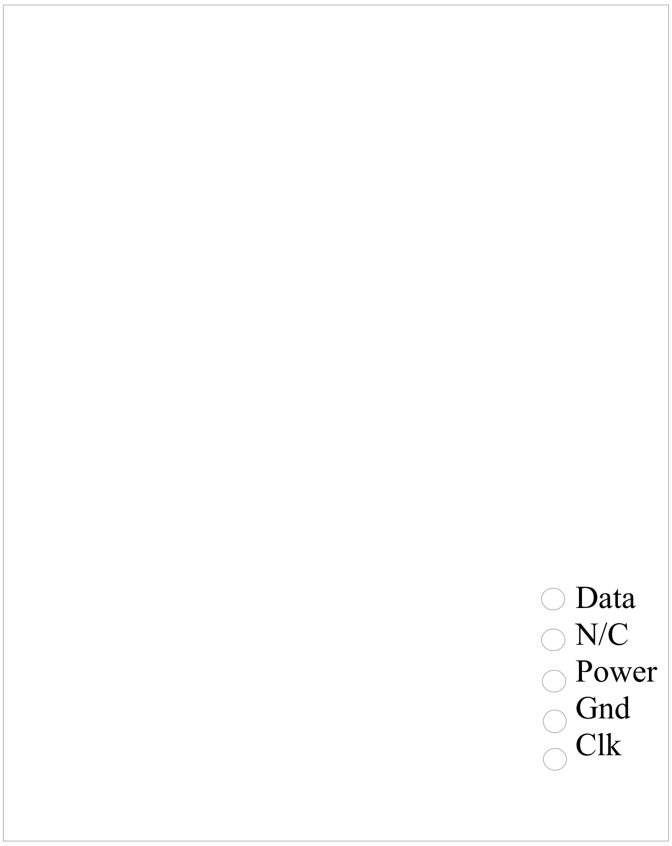
With the bootloader on the module, then you may use the “Update Multi” option (not Xmega) in the maintenance mode menu. Later revisions of the multi code detect a flash attempt and drop out to the bootloader. With earlier revisions of the multi code, then press and hold the “bind” button on the module, at power on, to force the bootloader to run.
In theory, it should be possible to use a FTDI device (not tested) to flash the module, but you will need a resistor in the Tx signal from the FTDI device if it outputs 5V.

WARNING: The Vcc voltage must not exceed 3.3V, and the Clk and Data signals are also limited to 3.3V.

It is likely easiest to use a separate power source, this connects to the Power and Gnd connections of the module connector. I used a 4-cell, NiMh pack on the module connector. Doing this makes use of the module's own 3.3V regulator. It is possible to use the power from the pin in the Tx module bay.

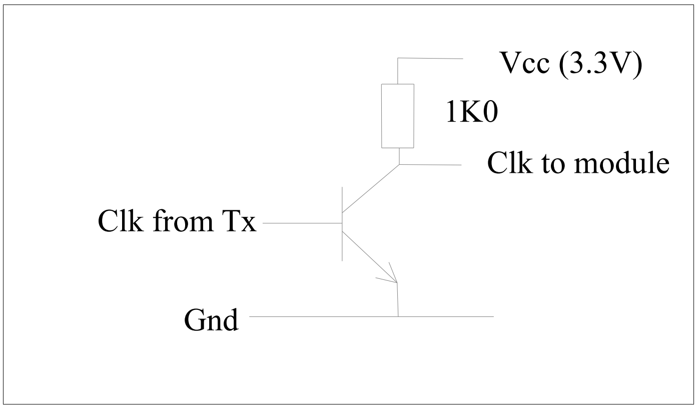
The connections to the transmitter are different depending on what transmitter you have, make sure you use the correct connections.

Module Bay connections for **9XR-PRO** and **9Xtreme**:



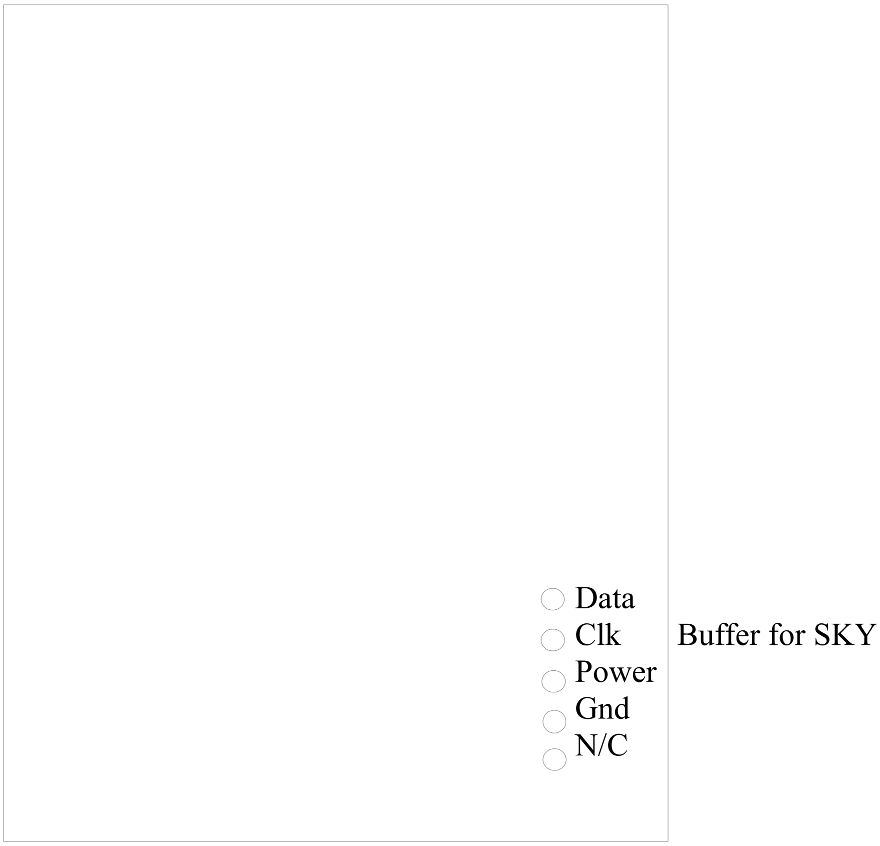
Buffer for 9Xtreme

Note that the Power connection is battery voltage, do NOT use this directly to the 3.3V connection on the module.
If using the 9XR-PRO then the module bay pins connect directly (Data, Clk and Gnd). If using the 9Xtreme, Data and Gnd connect directly, but you need the following transistor circuit to buffer the Clk signal:

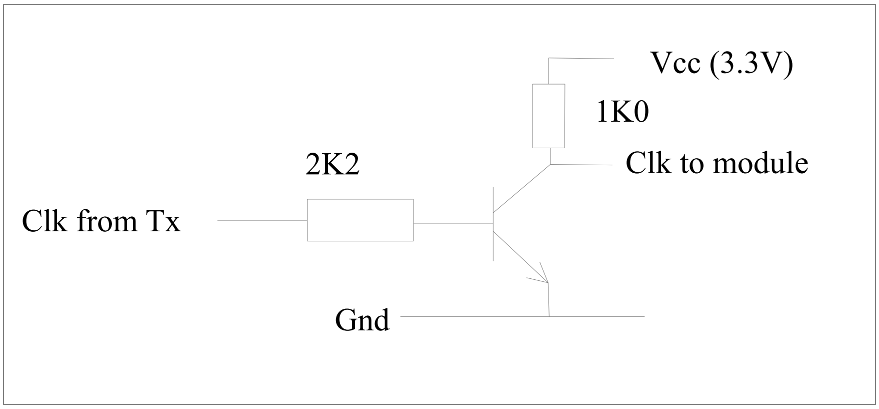


Note that there is a resistor internal to the 9Xtreme, so you don't need a resistor in the base of the transistor (any general purpose NPN type). The resistor actually stops the 9Xtreme from driving the signal fully, which is why the transistor is needed.

Module Bay connections for **AR9X** and **SKY**:



Note that the Power connection is battery voltage, do NOT use this directly to the 3.3V connection on the module.
If using the AR9X then the module bay pins connect directly (Data, Clk and Gnd). If using the SKY board, Data and Gnd connect directly, but you need the following transistor circuit to buffer the Clk signal:



Note that the SKY board has a real RS232 driver, so you need a resistor in the base of the transistor (any general purpose NPN type). The driver outputs +/- 6V which is why the transistor is needed.

For both the SKY and AR9X boards, the the COM1 output signal must be wired to pin 2 (one down from the top) in the module bay, and the back board needs the ground connections (on both sides) removed from this pin. On the AR9X board this signal is a 3.3V signal driven low by a transistor and high by a pullup resistor. The SKY board has a RS232 driver.

Updating.

In ersky9x, there are two new options in "maintenance mode" for updating the module. Use “Update Xmega” if connecting using the PDI programming connections. Use “Update Multi” if you have the bootloader on the module.

Using “Update Xmega”, you will need to:
1. Flash the new firmware to the radio (9XR-PRO or 9Xtreme).
2. Copy the MultiOrange.bin file to the firmware directory on the SD card (then power the radio off).
3. Connect the 3 connections from the module bay pins to the places on the Orange module.
4. Start the radio in maintenance mode (hold the two horizontal trim switches APART while powering on).
5. Power the Orange module through the main 5-way module socket.
6. Choose "Update Xmega" on the radio.
7. Select the MultiOrange.bin file (LONG MENU press).
8. Confirm the operation (LONG MENU press).
The module should then be flashed with the firmware, you will see a progress bar, and it takes only around a second. If it takes longer than this, then it probably didn't work (there is no error checking in at present).

Now disconnect the module and replace it in the module case.
You are now ready to plug the module into the radio and start using it.

Using “Update Multi”, you will need to:
1. Flash the new firmware to the radio (9XR-PRO or 9Xtreme).
2. Copy the firmware to the firmware directory on the SD card (then power the radio off). This may be either a .bin file or a .hex file.
3. Plug the module into the radio.
4. Start the radio in maintenance mode (hold the two horizontal trim switches APART while powering on).
6. Choose "Update Multi" on the radio.
. . . .

On the protocol menu, set the protocol to MULTI, and the Type to either DSM2 or DSMX as required. Then set the option value to:
option = number of channels and frame rate:

- 0 : 4 channels @22ms
- 1 : 5 channels @22ms
- 2 : 6 channels @22ms
- 3 : 7 channels @22ms
- 4 : 4 channels @11ms
- 5 : 5 channels @11ms
- 6 : 6 channels @11ms
- 7 : 7 channels @11ms
- 8 : 8 channels @22ms
- 9 : 9 channels @22ms
- 10 : 10 channels @22ms
- 11 : 11 channels @22ms
- 12 : 12 channels @22ms

When you power on, the LED on the module should flash once per second while the splash screen is showing, then go on solid when the radio starts to sends the MULTI protocol data. The MULTI code doesn't drive the beeper currently.

To bind, there is a BIND option on the menu, use this rather than the button on the module.

To enable telemetry, set the “Usr Proto” in the telemetry menu to DSMx.

26-Apr-2016