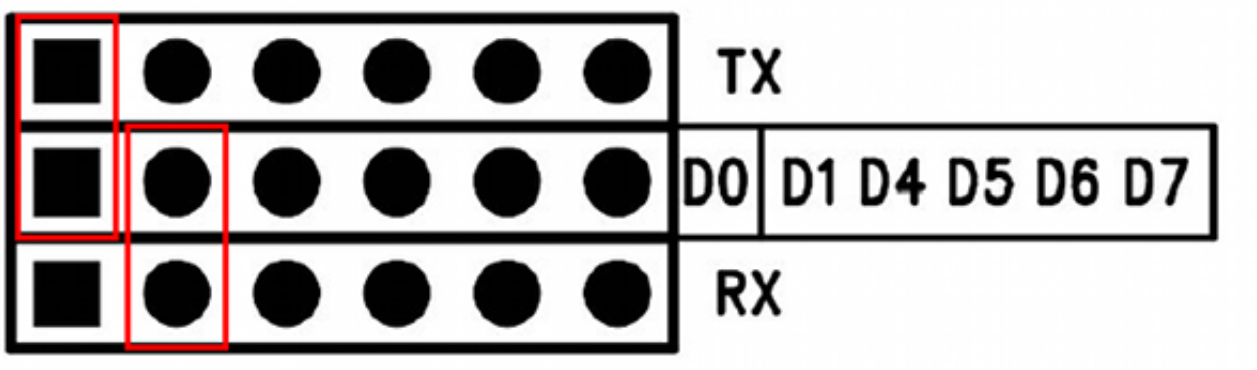
The Missing BT Shield Datasheet

The official ITead Studio BT Shield v2.2 Datasheet is missing lots of information (at least I was confused). So this web page is my attempt to summarize the missing bits.

Understanding the UART interface.

To me (who came to this totally unfamiliar with arduino), it wasn't at all obvious how the different pieces fit together to talk to one another. It turns out the TX/RX pins on pretty much everything are connected together one way or another, so the first rule is: Never try to use them on more than one device at a time.

When the bluetooth shield is jumpered like this:

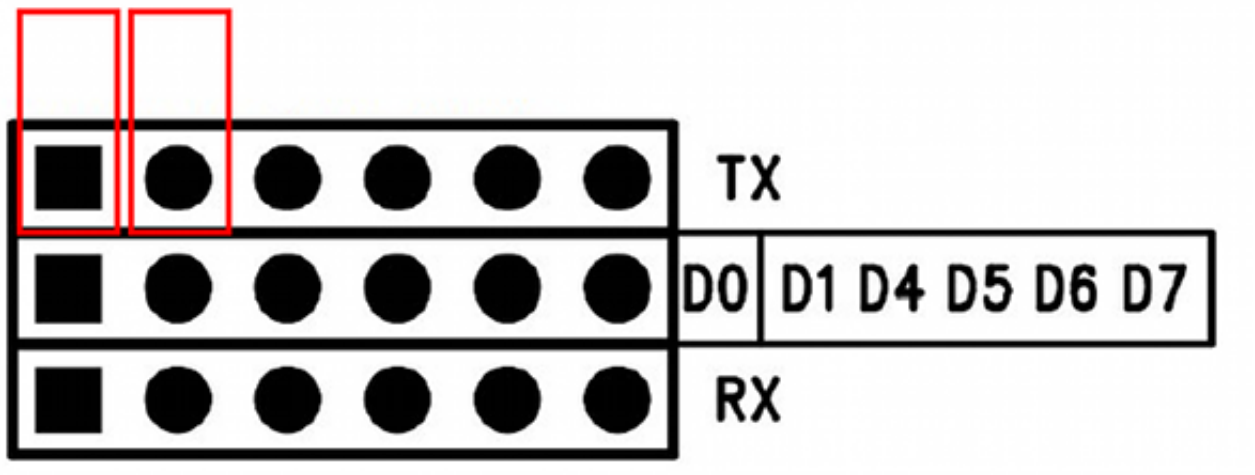


Then the serial data lines from the bluetooth module will be sending and receiving with the TX/RX uart interface on the arduino chip. This is the mode you normally use to do bluetooth communications with the arduino from some external bluetooth device such as a smartphone that talks over a serial bluetooth profile (RFCOMM).

Also note: The baud rate the bluetooth module is set to use needs to be the baud rate the arduino uses to initialize the serial port, otherwise they aren't going to talk well together.

But note that the USB interface chip is also connected to these same pins, so don't be trying to talk over the USB when you have the jumpers set this way, because that will simply result in nonsense. Leave the USB out of it when jumpered like this.

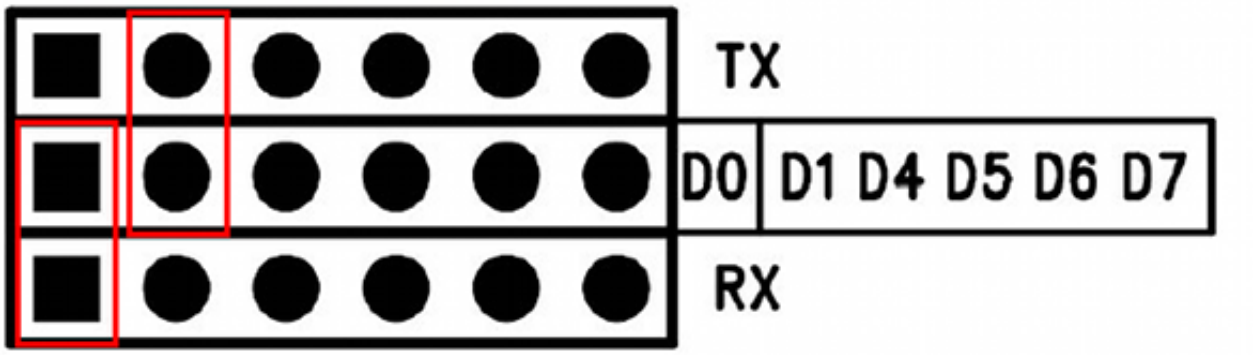
To go back to talking over USB, I need to remove the bluetooth shield from the equation by powering off and arranging the jumpers like so:



That just uses the pins as a place to store the jumpers, but not have them connecting anything, and it makes it safe to use the USB serial interface for comunicating with the arduino.

Command Mode

When you need to talk to the bluetooth module directly to issue the **AT** commands described in the official datasheet, then you need a different set of jumpers:



You also have to set the switch from the side labeled **DAT** (where it should be in all other modes) to the side labeled **CMD**.

This will connect the bluetooth module directly to the USB serial interface where you can communicate with it using a terminal program. It is important to note that in this mode the baud rate it uses to talk to the terminal is always 38400. Setting the baud rate with an **AT+UART=** command only sets the rate at which it communicates when talking to the arduino, it has nothing to do with this command mode.

Even more important to note is that these jumpers connect both the arduino and the bluetooth module to the USB serial interface. Therefore it is vitally important that the arduino be running a sketch that does not use the serial interface, otherwise you'll have another fine mess on your hands. I usually upload the standard example **Blink** sketch before I try to go into command

mode.

After configuring the bluetooth module in command mode, be sure you remember to set the switch back to **DAT**.

Bluetooth Upload

I had a lot of problems with trying to upload over bluetooth until I figured out the baud rate stuff above. Once I got the bluetooth module set to use 115200 by using the command mode, then the bluetooth module is talking the baud rate the bootloader expects.

There is still no way for the bluetooth module to cause a reset, so I have to do that manually, but it is easier to get the timing right for the manual reset when you run avrdude from the command line or via a Makefile (like I figured out in Remote Software).

My procedure is now to run **make clean build** which gets the new microcode ready to upload, then, without hitting enter, type the command **make upload** in a terminal. Now I can hold down the reset button on the arduino then hit enter in the terminal at the same time I release the reset button. So far that procedure has worked flawlessly.

Naturally, I need to make sure the Makefile is configured to use **/dev/rfcomm0** instead of **/dev/ttyACM0**.

Pin 2

Not mentioned anywhere in the official datasheet is the fact that the **Status** led is apparently connected to arduino pin 2, so if you have any plans to use pin 2 for something, forget 'em :-). I guess the theory is that you can monitor the blink rate to tell if the bluetooth module is connected to something, but that seems like a moderately useless thing to tie up one of the only two pins that can generate external interrupts.