SDRSharp Physical Controls

This page gives an example of using a rotary encoder and Arduino to control the frequency in SDR#.

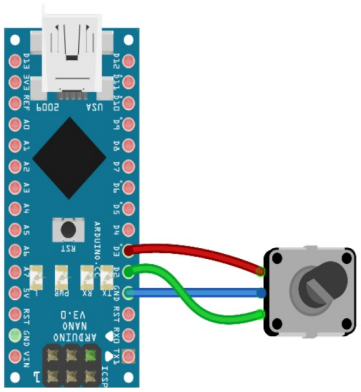
As the SDRSharp Net Remote plug-in can now read commands from a serial port it's relatively easy to control SDR# with buttons, potentiometers and rotary encoders (to name a few) with the help of an Arduino or your favourite microcontroller.

All you need is an Arduino (I used a Nano) and a rotary encoder.

I'll assume you have SDRSharp installed along with the Net Remote plugin and the Arduino drivers (so it shows up as a COM port).

1. Wire up the encoder and Arduino

Nice and simple - only 3 wires needed:



Arduino	Encoder
D2	B
D3	A
GND	C

Swap the A & B wires if you want to change direction.

Some encoders have 2 extra pins for the shaft push switch - you can ignore them.

2. Upload the code to the Arduino

Download or copy and paste the sketch from GitHub and open it with the Arduino IDE.

You'll also need the Encoder Library to interface with the encoder - download it then add it to the Arduino IDE by selecting 'Sketch -> Include Library -> Add .ZIP Library...' from the menu.

The code is pretty simple:

Line 31: Include the Encoder Library so we can use it's functions
Line 34: Initialise the encoder on digital pins 2 & 3
Line 35: We'll use the 'oldPos' variable to track if the encoder has moved
Line 38: The 'frequency' variable stores the initial frequency, here it's set to 100MHz
Line 42: Open up the serial port and set the baud rate to 115200
Line 45: The 'loop' function is called over and over again so the main part goes in hereLine 47: Read the encoder position
Line 48: See if the encoder has been rotated since we last looked
Line 52-54: Send a command to change the frequency.
Line 53: Get the encoder position, multiply it by 1000 - so each step on the encoder changes the frequency by 1KHz

The command to change the frequency is in JSON and looks something like:

```
{"Command": "Set", "Method": "Frequency", "Value": 100000000}
```

Which translates to '**Set** the **Frequency** to **100000000**'. Nice and easy!

3. Give it a blast!

Plug in the Arduino and fire-up SDR#, in the control panel for Net Remote make sure you select the serial port of the Arduino and then put a tick next to 'Serial'.

You should now be able to twiddle your knob and see the frequency change in SDRSharp.

Problems?

Using the Arduino IDE open the serial monitor ('Tools -> Serial Monitor'), change the baud in the bottom right corner to 115200. If you rotate the encoder it should spit out commands, one per line.

The title bar of the serial monitor will show which port you are using, this should be the same in SDR#. Please note you can't use the serial port with more than one application at a time so either have SDR# or the serial monitor running, not both at once (unless you like error messages!).

Make sure SDR# is using a real radio, if the source is set to 'IQ File' or 'Other' the frequency can't be changed.

What next?

Use the digital pins to read button presses or the analogue pins to read the position of a potentiometer.

Rather than setting the starting frequency in code you can read it's value from SDR# on start-up. This way the frequency won't be reset the first time you move the encoder.
You can read the current frequency using the command:

```
{"Command": "Get", "Method": "Frequency"}
```

Which responds with something like:

```
{"Result":"OK","Method":"Frequency","Value":101500000}
```

To make life easy I'd use one of the JSON libraries to extract the frequency.

Categories:
Examples and Tutorials

Click to view comments

Click to add a comment

Cell Map

Find the location of cell transmitters

UK Quake

Map UK earthquakes

GPX Logger

A GPS data logger