readme.txt

1 – The device is shipping without firmware.

The device will keep entering “rescue mode” on boot until the firmware is installed.

- Connect power, either via the DC jack or using USB-C PD. Do not exceed the rated input voltage.
- The device will boot showing two RED leds, then two ORANGE leds, and finally two PURPLE leds. The static, purple leds indicate that you are in firmware update mode.
- Connect an ethernet cable. The device will begin downloading the firmware, which is less than one GB. The purple leds will begin flashing alternatively. Every flash is a 1% progress of the current step. There are several steps.
- Alternatively, you can download the firmware update file < http://bin.rfnm.io/rfnmblue.swu > into a usb stick and plug it in.
- When the firmware update is successful, the two leds will become static BLUE. All is done, reboot the device by disconnecting power.

To upgrade the firmware again, flip the boot switches to *Rescue* and repeat the steps above. A better firmware update method will come.

You don’t have to worry about software updates bricking the device, there are simple recovery methods in that remote eventuality.

2 – Getting Started

After the normal boot process, the two leds are supposed to be ORANGE and BLUE. When in this state, you can use a USB 3.0 SuperSpeed Type-C cable to connect the device to your computer.

The device will expose a USB drive to your Windows machine. Inside, you will find a portable version of SDR++ that should let you start playing within seconds.

The ethernet jack is setup to autoconfigure itself using DHCP. There is a default SSH server running, which lets you connect to the i.MX. The default user is “root”, with an empty password. You can also connect using UART by plugging in an adapter set to 115200 baud to UART 2 on the side of the board.

3 – Playing around a little bit

dmesg | grep RFNM

> Gives you an idea of what happened during boot, info on daughterboards, etc.

echo on > /sys/class/i2c-dev/i2c-0/device/0-0058/rfnm_ext_ref_out

> Enables the 10 MHz reference output. You can clock the device from an external 10 MHz source by feeding it to the REF_IN port, it's automatic.

tree /sys/kernel/rfnm_*

> Sysfs configuration interface, also replicated on *github/rfnm/librfnm*

watch -n 0.2 cat /sys/kernel/debug/rfnm/stream_status

> Live data status information. More fun while streaming data.

cd /sys/class/leds/rfnm_wsled/device/ && echo 250 > chain0_led0_r && echo 1 > chain0_apply

> Set the red component of the first LED of chain0 to max brightness and apply.

ls /rfnm/scripts/

> Some scripts used by the system. For example, you can enable the USB-A port as a host with *enable_usb-a* or start streaming with *sysfs_demo*.

echo 153 > /sys/class/i2c-dev/i2c-0/device/0-0050/rfnm_set_dcs_freq && reboot

> Change the main sampling frequency from 122.88 to 153.6 MHz, which requires two usb connections (the second one acting as a data boost).

5 – Soapy and GnuRadio

There is an initial Soapy implementation on our github. CMake should build it just fine. The windows version of the same driver and a sample GnuRadio block diagram also ships in the same USB drive folder exposed by the system at boot.

6 – Answering your questions

Please join our Discord. I will be helping everyone as much as I can, but if the help is public, hopefully I won't need to repeat the same things as many times. You can find the Discord link at the bottom of our website.