GNU Radio's Control Port

Tom Rondeau

trondeau.com

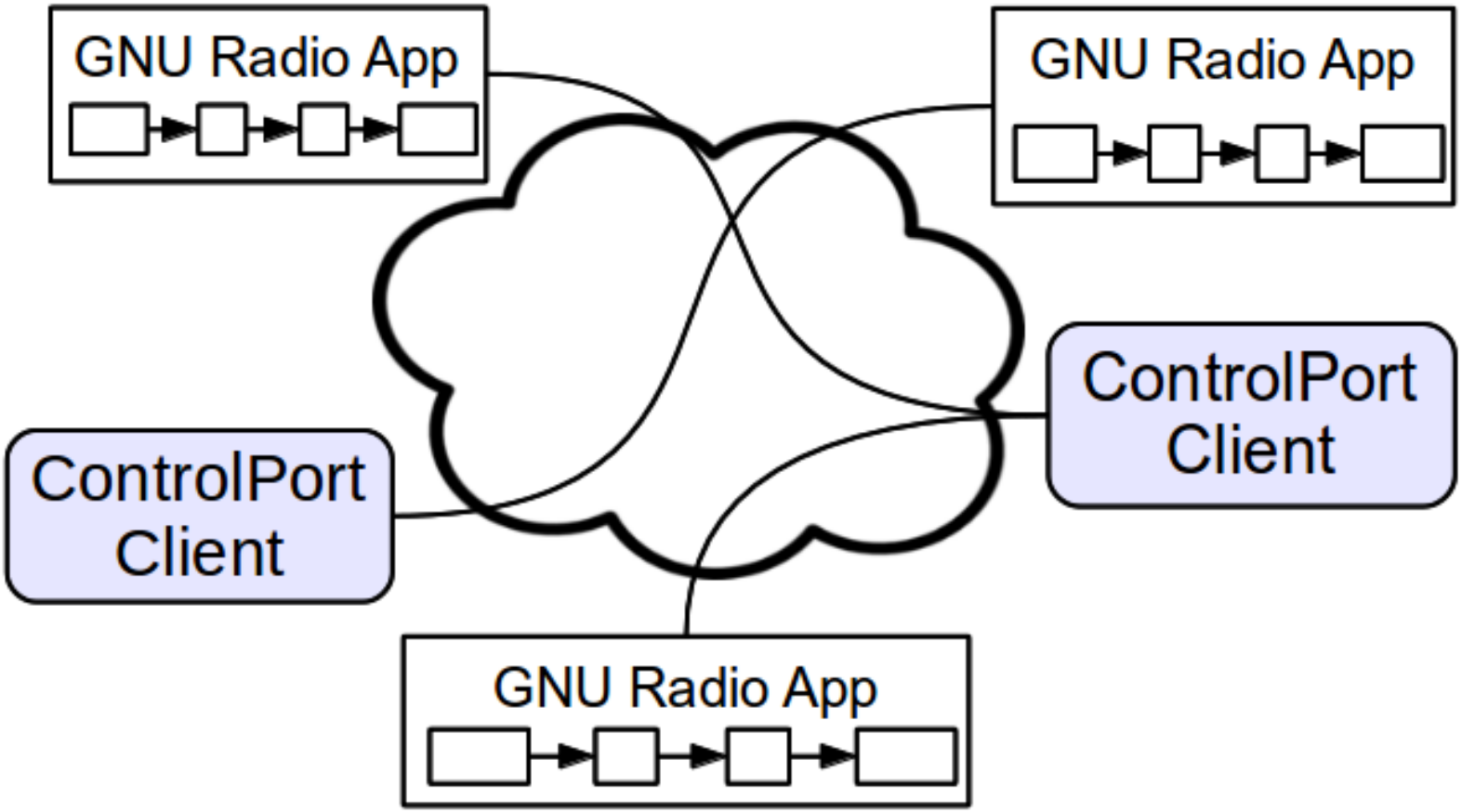
2013-10-03

GNU Radio ControlPort (gr-ctrlport)

Remote control and visualization.

- Use of ControlPort to enable debugging without requiring extra debug streams.
- Based on a robust and secure backend (ICE)
- No additional CPU usage while no monitoring is occurring.
- Can connect multiple remotes to same GNU Radio application.
- Can also have single ControlPort app control multiple GR apps.
- Easy to use IceGrid to enable a grid of applications.

Control Port Concept



Interfaces

ControlPort defines interfaces to GNU Radio.

- Described in **gnuradio.ice**.
- Uses Slice language.
- Defines data types and structures for moving data.
- ControlPort Interface is the main class:
  - set –> change a GR block's variable.
  - get –> get the current variable's value.
  - properties –> a map of all available variables.

Config File Control

Controlling ControlPort

- Section [ControlPort] in **gnuradio-runtime.conf**.
  - **on**: Can be On/True/1 or Off/False/0 (defaults to off).
  - **edges_list**: toggle exporting a list of flowgraph edges.
  - **config**: the middleware-specific configuration.
    - ICE uses its own syntax.

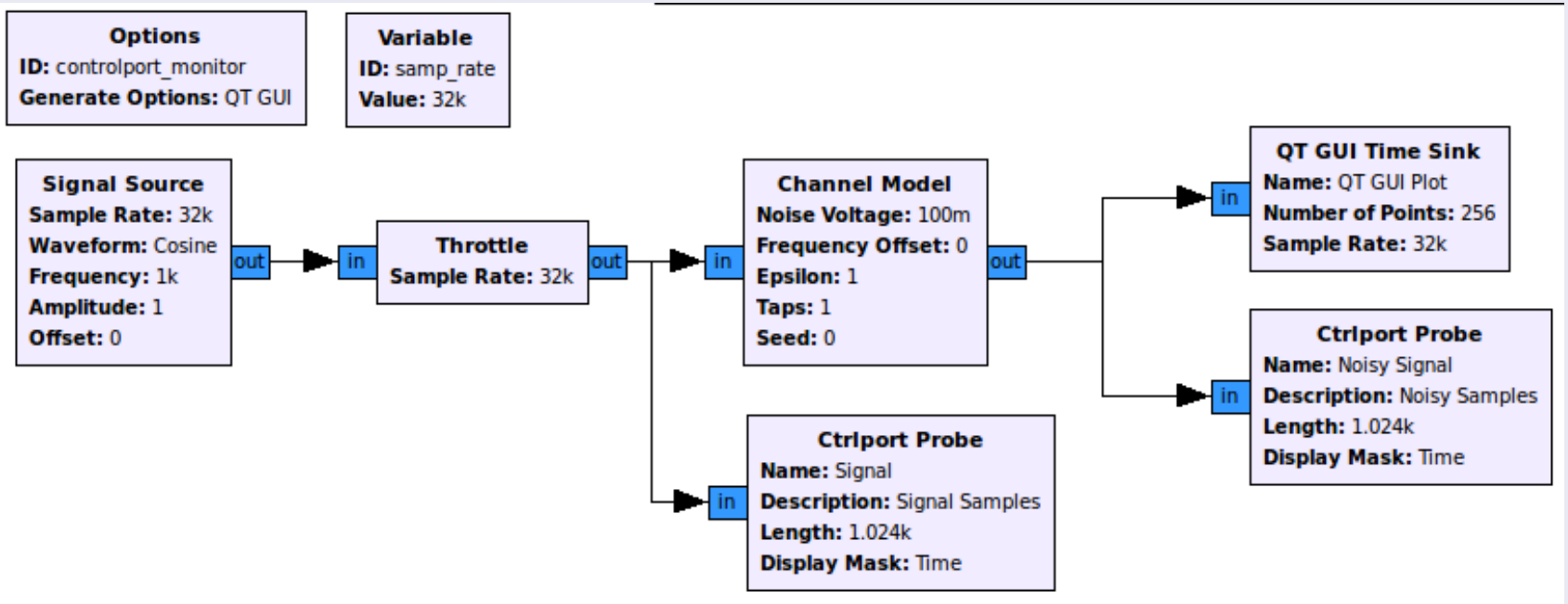
Tips on configuration files

- Can locally override any options with a personalize config file:
  - ~/.gnuradio/config.conf
  - Just add the section name and any variables to set for you.
- Can also set environmental variables:
  - GR_CONF_<section name>_<option name> = <value>

ControlPort Probes

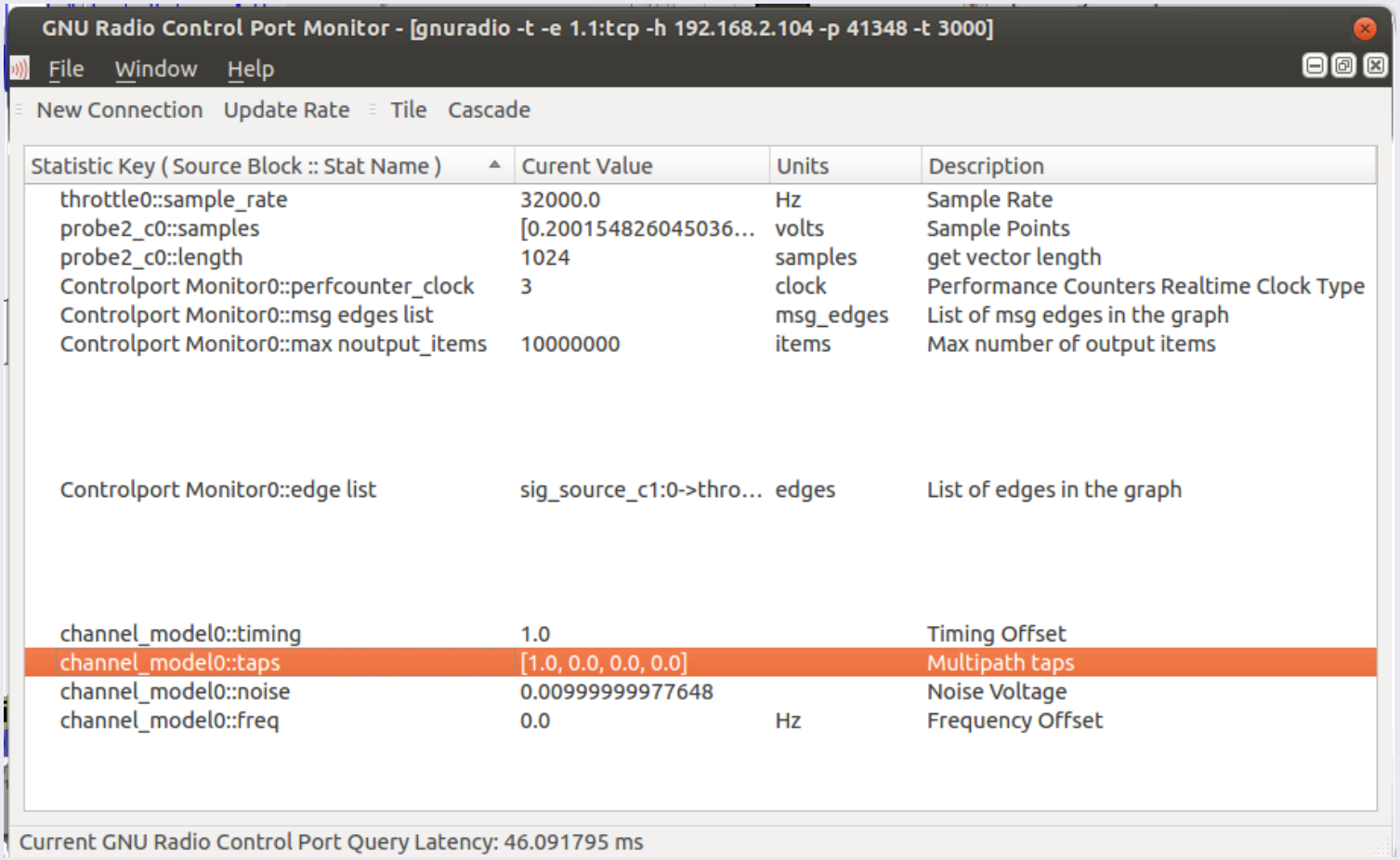
GNU Radio sink blocks.

- Automatically export vectors of data.
- Easy to place into a flowgraph and monitor externally.



gr-ctrlport-monitor

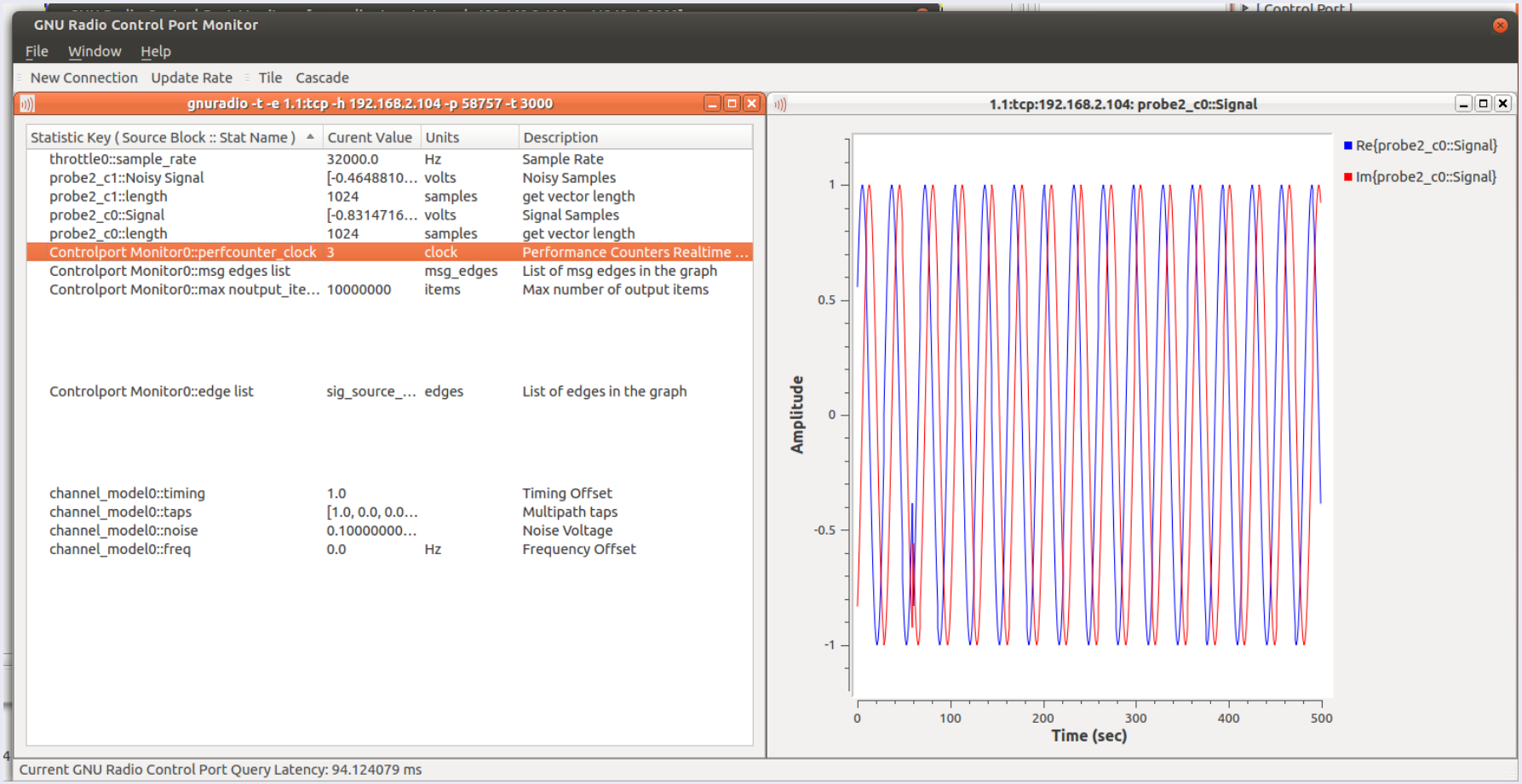
Command-line tool installed with GNU Radio.



- In GRC under *Control Port* as *CtrlPort Monitor*.

gr-ctrlport-monitor Interaction

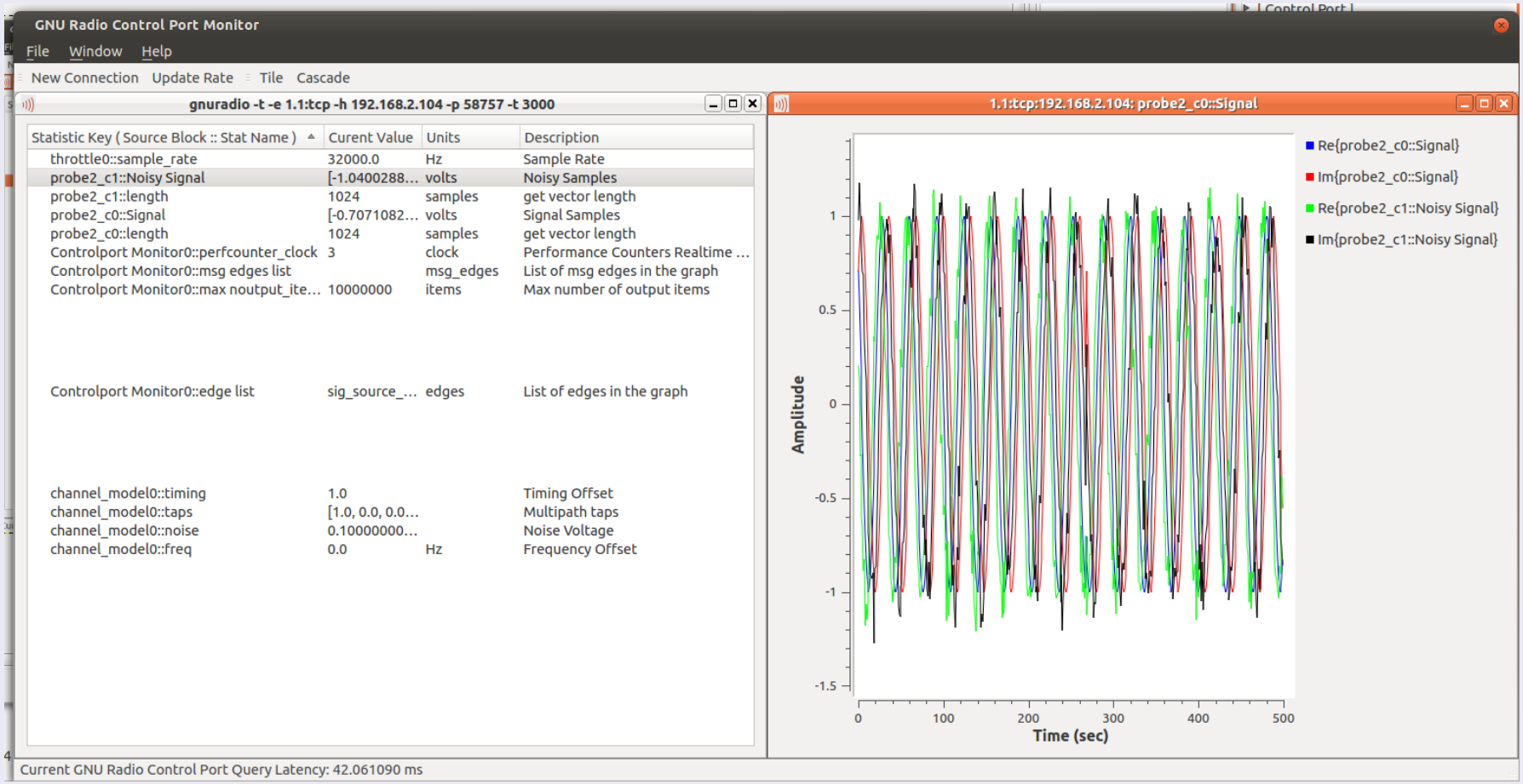
Double-click will pull up a default plotting tool.



- Interfaces describe what this default is.

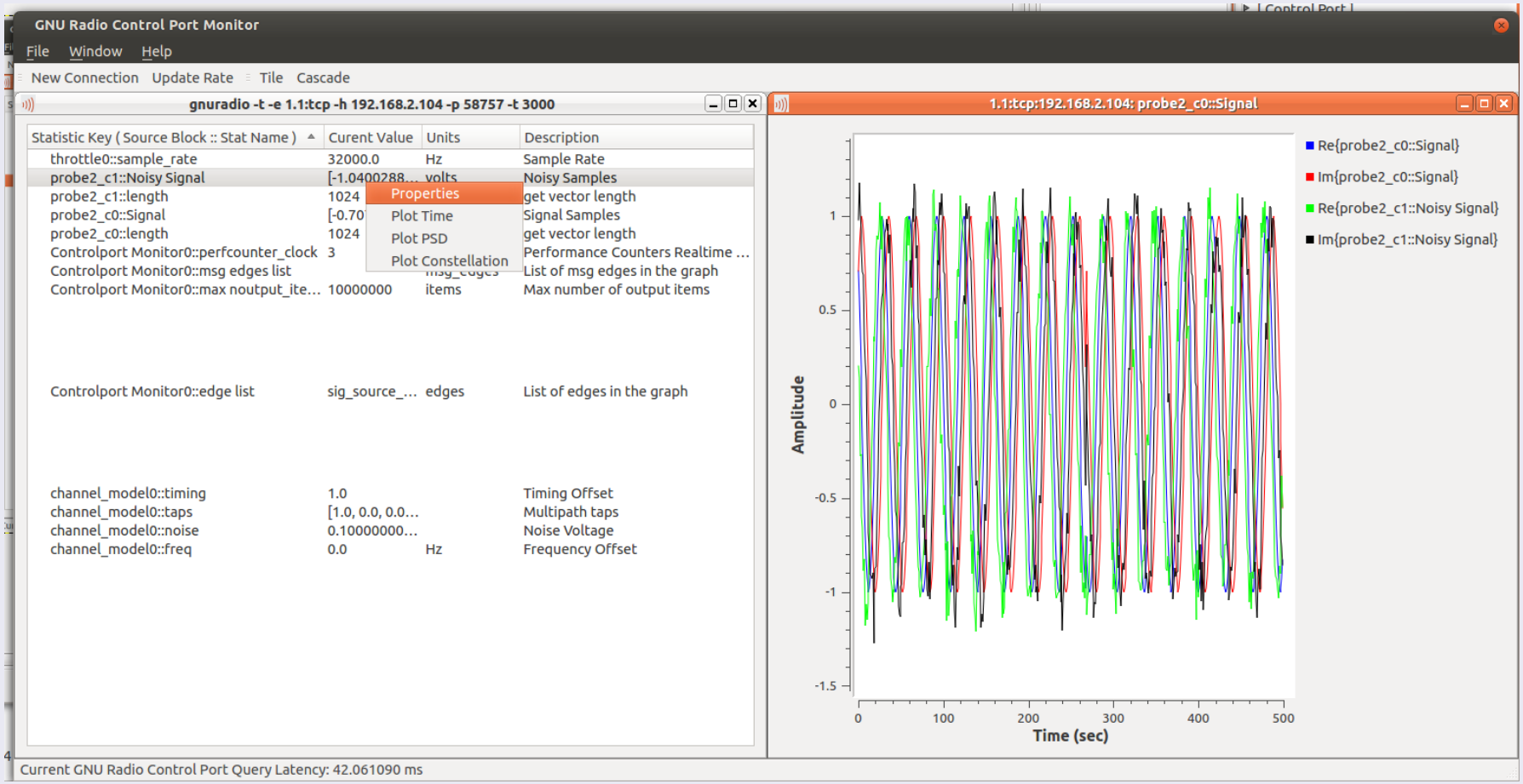
gr-ctrlport-monitor Interaction

Drag-and-drop other data streams on top of the existing plots.



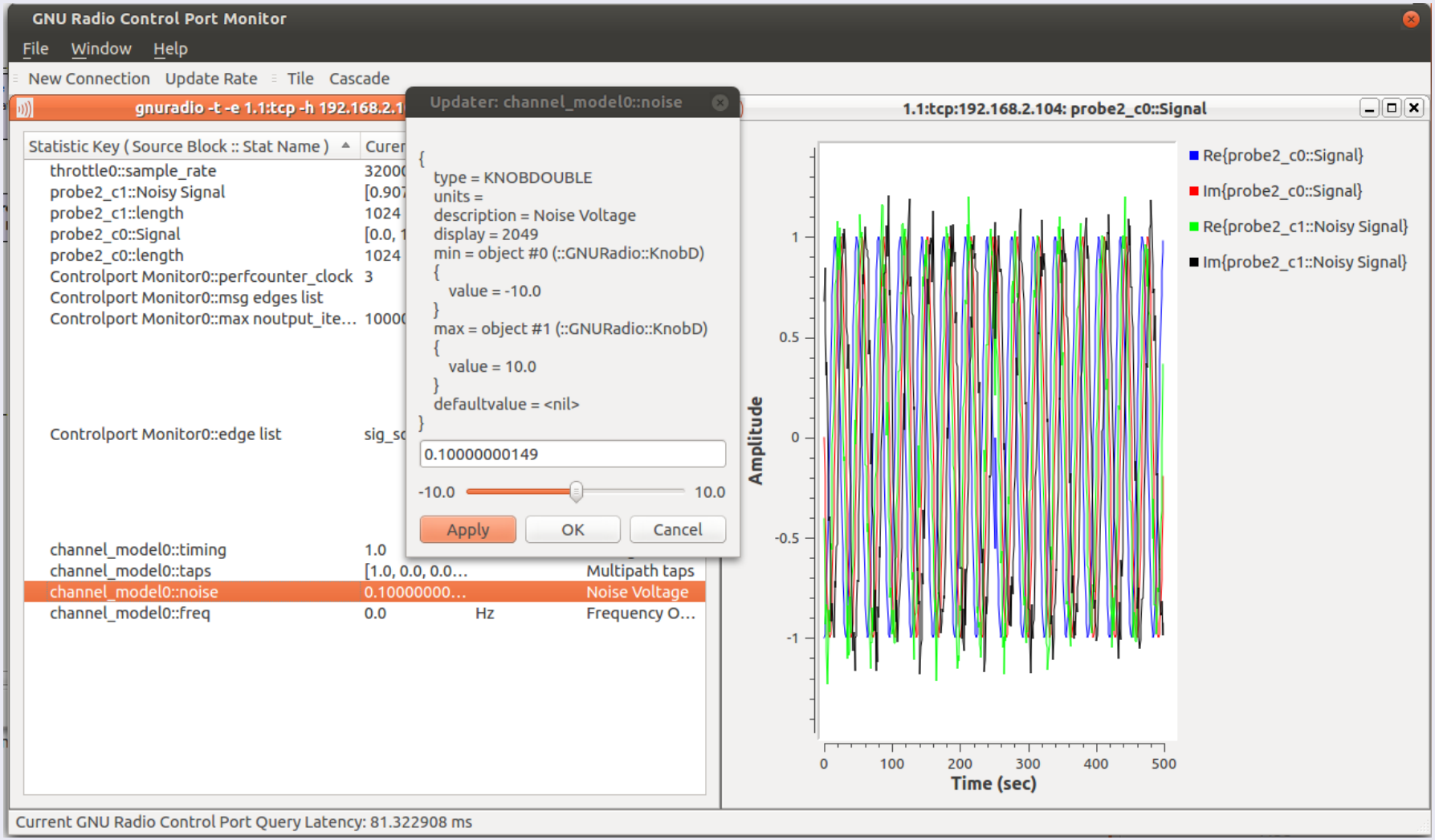
gr-ctrlport-monitor Actions

Right-click for other plot types and actions.



gr-ctrlport-monitor Set

Can set values if interface defined.



Creating an Interface

Methods built in to `gr::block`

- `d_rpc_vars`: simple container for any variable's setter/getter.
  - We never have to touch this; just know it's there.
- Register a new variable's set/get by adding it to `d_rpc_vars`.
  - Conviniently wrapped this as `add_rpc_variable(sptr)`.
- Add these calls to a `setup_rpc()` function in the block.
  - This is called for each block when start() is run.
  - Overload it in your block's class.
  - Will cover syntax of creating the sets/gets later.

Exposing variables

Declaring a 'get'

```
add_rpc_variable(
  rpcbasic_sptr(new rpcbasic_register_get<channel_model, double>(
          alias(), "noise",
          &channel_model::noise_voltage,
          pmt::mp(-10.0f), pmt::mp(10.0f), pmt::mp(0.0f),
          "", "Noise Voltage", RPC_PRIVLVL_MIN,
          DISPTIME | DISPOPTSTRIP)));
```

Exposing variables

Declaring a 'set'

```
add_rpc_variable(
  rpcbasic_sptr(new rpcbasic_register_set<channel_model, double>(
          alias(), "timing",
          &channel_model::set_timing_offset,
          pmt::mp(0.0f), pmt::mp(2.0f), pmt::mp(0.0f),
          "", "Timing Offset",
          RPC_PRIVLVL_MIN, DISPNULL)));
```

What the set/get format means:

- set the `rpcbasic_register_{g,s}et<T, Td>`
  - T: class type
  - Td: data type of the variable
- alias(): the block's alias and method for getting the object.
- "timing": a name for the interface.
- &channel_model::**set_timing_offset**: function pointer to the get/set routine.
- Next three are PMTs to suggest min/max/default value.
- "": units, if any.
- "Timing Offset": a description.
- RPC_PRIVLVL_MIN: a privilege level required to access.
- DISPNULL: hints to the ControlPort client when plotting.

Display Options are a bit mask

Plot Types

- DISPNULL: Nothing specified.
- DISPTIME: Time-domain plot.
- DISPXY: XY or constellation plot (complex only).
- DISPPSD: PSD plot.
- DISPSPEC: Spectrogram plot.
- DISPRAST: Time raster plot (non-complex only)

Plot Options

- DISPOPTCPLX: Signal is complex.
- DISPOPTLOG: Start plot in semilog-y mode (time domain only).
- DISPOPTSTEM: Start plot in stem mode (time domain only).
- DISPOPTSTRIP: Run plot as a stripchart (time domain only).
- DISPOPTSCATTER: Do scatter plot instead of lines (XY plot only).

Adding setup_rpc to a block

Declare #include <gnuradio/config.h> in impl file:

```
void
bar_impl::setup_rpc()
{
#ifdef GR_CTRLPORT
  add_rpc_variable(
    rpcbasic_sptr(new rpcbasic_register_get<bar, float>(
      alias(), "multiplier",
            &bar::mult,
      pmt::mp(-100.0f), pmt::mp(100.0f), pmt::mp(0.0f),
      "", "Multiplier", RPC_PRIVLVL_MIN, DISPTIME | DISPOPTSTRIP)));
  add_rpc_variable(
    rpcbasic_sptr(new rpcbasic_register_set<bar, float>(
            alias(), "multiplier",
            &bar::set_mult,
            pmt::mp(-100.0f), pmt::mp(100.0f), pmt::mp(0.0f),
            "", "Multiplier", RPC_PRIVLVL_MIN, DISPNULL)));
#endif /* GR_CTRLPORT */
}
```