“You have turned on the future!”

Updates from Ettus Research R&D

Martin Braun
Senior Git Wrangler, Ettus Research
GRCon ‘17
N310: Not just a respin

- A bit of E310, a bit of X310, and many other things
- It’s 2017, and we have many more requirements
  - Feedback from E310, X310, and RFNoC users went into the software + hardware design
- If you enjoy boredom, you’d hate working for Ettus
Moving to more systems: RASM

- Enter the marketing terminology
- Systems of many devices are worthless if there’s no good way to manage them
- Field upgrades, fan controls, remote diagnostics, remote self-tests, close-to-hardware APIs, monitoring, ...
Multi-Device Management

- RASM is not just a selling feature
- Debugging multi-usrp setups makes us creative…
- In the early stages, ghetto-RASM may be sufficient…
  - $ cat iplist | xargs -L 1 -t -n 1 -Iusrpip -P 1 ssh root@usrpip ls
  - $ tmux setw synchronize-panes on
Remote Deployment: Mender.IO

- Solution to remote updates of many devices
- Disclaimer: We will (try) not (to) force tools onto
More devices? Smarter devices!

"The IQ of a mob is the IQ of its most stupid member divided by the number of mobsters." -- Sir Terry Pratchett

- More and more, clusters of many USRPs are becoming prevalent
- Classic separation of UHD + FPGA is no longer sufficient
- Let the embedded OS do some of the heavy lifting
Network Mode

- Number two feedback for E310 was the lack of N200-like mode ("network mode")
- But of course, running UHD embedded is also appealing
- ARM Cortex-A9 is not super fast, but SFP+ ports must not be slowed down
- ...and also, we don’t want two versions of UHD per device
MPM

- Boil it all down, and out comes…
- Module Peripheral Manager
- Software local to device
- Written in Python, C, and C++ (whichever version we prefer)
MPM

User App - GNU Radio

RPC

API

CHDR

UHD

MPM

FPGA

AXI Interconnect

RPC

API

CHDR

UHD

MPM

FPGA
▪ Good for device, good for developers
▪ Move many problems outside of UHD (and confines thereof)
▪ We can use Python
▪ Command-line interface
▪ Better separation, less dependencies during development

```
$ python3 ./mpm_shell.py 192.168.30.2 --claim
Attempting to connect to 192.168.30.2:49601...
Connection successful.
Getting methods...
Added 46 methods.
Quering device info...  
Claiming device...
310A849 [C]> db_0_
db_0_check_deframer_status  db_0_is_initialized
db_0_get_serial  db_0_send_sysref
db_0_init_adcs_and_deframers  db_0_shutdown
db_0_init_jesd_core_reset_adcs  db_0_update_ref_clock_freq
310A849 [C]> db_0_
```
UHD Updates

- C++11 coming to UHD with next major release
- Logging API
- 3.11 release, to replace 3.10 release cycle, 1-2 months out
- 3.9 LTS branch will live on for a while

```cpp
// Preload all the tasks (they might start running on emplace_back)
for (const auto &block_ctrl: _rfnoc_block_ctrl) {
    auto rpc_block_id = block_ctrl->get_block_id();
    if (has_block<uhd::rfnoc::rpc_block_ctrl>(rpc_block_id)) {
        const size_t mboard_idx = rpc_block_id.get_device_no();
        auto rpc_block_ctrl =
            get_block_ctrl<uhd::rfnoc::rpc_block_ctrl>(rpc_block_id);
        auto rpc_spotr = _mb[mboard_idx]->rpC;
        task_list.emplace_back(std::async(launch_policy,
            [rpc_block_id, rpc_block_ctrl, &block_args, rpc_spotr](){
                rpc_spotr->block(mboard_idx, block_args);
                return rpc_spotr->block(mboard_idx, block_args);
            }));
    }
}
```
No, there’s not just a Python API. There’s two Python APIs!
- Ettus Research: Boost.Python based (Mirrors C++ API)
  - Hosted on Ettus github
- NRL: ctypes-based, uses ffi (Mirrors C API)
  - https://github.com/pysdr/pysdruhd
- No SWIG was involved in either Python API

Source: Stack Overflow
Ettus Python API

- Current state: Public feature branch github
  - $ git checkout python-api
  - We’re collecting feedback on our issue tracker

```python
def main():
    args = parse_args()
    usrp = uhd.usrp.MultiUSRP(args.args)
    num_samps = int(np.ceil(args.duration*args.rate))
    if not isinstance(args.channels, list):
        args.channels = [args.channels]
    samps = usrp.recv_num_samps(
        num_samps, args.freq, args.rate, args.channels, args.gain
    )
    with open(args.output_file, 'wb') as f:
        np.save(f, samps, allow_pickle=False, fix_imports=False)

if __name__ == "__main__":
    main()
```

[ RFC ] Python API #105

First of all, I love having a python API and I can’t wait until it’s shipped by default.
Python API + Jupyter

- Run your signal analysis directly inside your notebook!
- Convenience methods make it easy to pull samples in and out of USRPs from Jupyter
- Complements on-line, real-time processing in GNU Radio
- Will make you look so awesome in the classroom!
What happened to my issue?

- Ever posted a bug on our public github issue tracker?
- You might get something like this:

  ![Comment from mbr0wn](image)
  
  You're right, they should be removed. I'm moving this to our internal tracker (and thus, closing).

- So... what happens now?
A bug’s life

Intermediate public feature branch

- Continuous Integration
- Code Reviews
- Internal Tests
Final Word of Advice

- If you want to develop radio hardware...
- ...use GNU Radio. You won’t regret it.
- Great visuals
- Plenty of DSP blocks
- Easy to modify
Thank you all!

- ...for being here this week
- ...for contributing to GNU Radio
- ...for asking great questions!