PMm^2 Annual Meeting
2009-03-02 Orsay

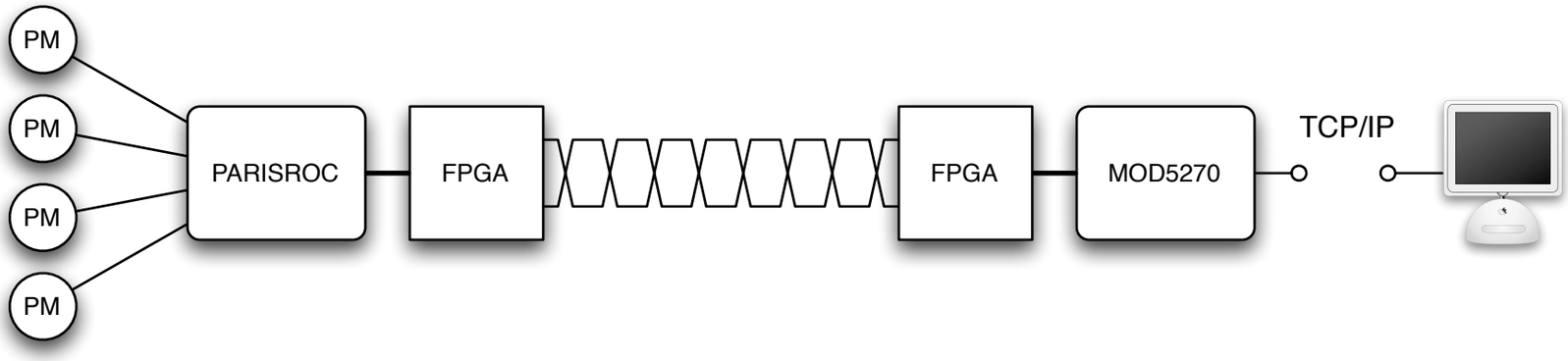
A DAQ for PMm^2

and beyond ...

Kael Hanson
Université Libre de Bruxelles

PMm^2 Annual Meeting - 2 March 2009 (Orsay)

Deliverables



- Client application for configuration, control, and readout of PMm^2 prototype array. It is envisioned that this system will be used for:
  - Debugging DAQ hardware (PARISROC, communication system)
  - Understanding the parameters of the system for future development (noise, PMT characteristics, ...)
- User documentation

Agile programming

- Experience with other large software projects indicates that it is possible to overdesign software
- Typical problem with this type of software is that requirements are not well understood until the software has been used over-and-over.
- Agile programming seeks to get something working as quickly as possible then features are added as experience with system grows.
- Nonetheless, we have to start somewhere ...

Use cases to bootstrap first offering

1. User is using Windows / Linux / Mac OS X as computing host for interactive client.
2. Client connects to MOD5270 over network using TCP sockets.
3. User manually read and writes MOD5270 / PMm^2 registers at low-level.
4. User programmatically reads and writes MOD5270 / PMm^2 registers at low-level.
5. User has defined configuration states that he/she wishes downloaded into the apparatus from the client without having to manually re-enter them.
6. User wants collect data from one or more channels and capture the data on disk.

Choice of development language

- From UC#1: Need cross-platform development language that supports network programming. I'd like to use Java for the following reasons:
  - First, and formost, I have *lots* of experience here,
  - TCP streams are supported at core and are easy to program,
  - It is truly cross-platform, even Python has some annoying corners when it comes to Win / Unix.
  - Swing UI toolkit if we want to add GUI later
  - Library of useful DAQ routines from IceCube DAQ - sorting multiple streams of time-stamped data, for example
- This will require a minimum of software setup: a JRE is all that will be needed for runtime.

Manual or programmatic access

- Using Jython (Python on Java) or some other scripting framework for Java (e.g. JRuby) one may already start to build complicated scripts to achieve UC#3-6.
- I realize that users are not necessarily programmers: it will take some discussion with people who interact with the hardware to figure out what is wanted. I hesitate to jump too quickly on GUI until there is a more concrete usage pattern that emerges.

Real (simulated) session

```
>>> from be.ac.iihe.pmm2.comm import PMM2Box00
>>> box = PMM2Box00('localhost', 7332)
>>> box.writeConfigurationRegister([ x for x in range(16) ])
```


6

Status

- Simple messaging layer is coded (not tested against real device, but tested against simulated objects): this just 'formats' the messages properly into the encapsulation structure and sends over socket.
- I have several *MockBoxes* which (now, rather unintelligently) simulate the MOD5270 but allow me to snoop on traffic being sent and also allow unit tests with a real client object.
- This is about 500 lines of code and took 15 hours of logged working time to accomplish.
- Once I understand how people want to begin using I can provide
  - canned solutions (e.g. prompt-driven clients) for those who don't want to deal with a programming language
  - the libraries and documentation for how to build larger structures
- If this is going down wrong path, we can bulldoze all the way to the ground and start over. An agile programmer is not afraid to junk large sections of code when the prototype doesn't work.

7

Demo

8