Trigger and DAQ Status

Suerfu

SABRE General Meeting

October 4, 2017

Outline:

1. Overview of SABRE trigger and DAQ
2. Possible upgrade to DAQ software
3. Upgrade to full-scale experiment

Overview of SABRE trigger and DAQ

- Trigger requirements:
  - Multiple pairs of PMTs coupled to multiple crystal modules
  - 10 veto PMTs
  - For dark matter searches and potassium counting, only crystal coincidence is necessary

- ADC requirements:
  - Scalability - multiple crystal module, more veto PMTs in the full-scale experiment
  - Event rate is low - projected to be on ~ Hz level
  - Fast veto signal, slower crystal signal
    - Decay time ~10 ns vs ~230 ns

Overview of SABRE trigger and DAQ

- Decided (by Princeton) to use CAEN VME systems:
- Trigger - V1495
  - General-purpose VME board with Altera Cyclone II FPGA.
  - FPGA firmware is Implemented via VHDL.
    - 4 trigger modes : dark matter, liquid scintillation, veto, calibration
    - Trigger scheme is very simple (PoP resource usage < 0.1% on the chip)
  - V1495 has only 2 output LEMO terminal - TRIG signal from V1495 has to be chained in V1720
  - Trigger information has to be passed to V1495 via LVDS (low-voltage differential signaling).

- ADC - V1720:
  - 250 MS/s, 12-bit resolution - enough for pseudocumene & NaI
  - Daisy-chainable - scalability proved in DarkSide
  - All channels and boards share the same bandwidth of ~ 85 MB/s

Overview of SABRE trigger and DAQ

- Status:
  - Trigger and DAQ has been set up in SABRE PoP temporary site in Hall B in March 2017
- The system was tested with 8 PMTs concurrently operating
  - a (retired) BGO crystal was suspended in the vessel
  - BGO light yield was too poor
  - In the first overnight run, a muon accidentally hit the crystal and gave a beautiful pulse
  - However earlier in the day, a worker tripped over the cable, breaking a connector on ADC
- Further tests going on:
  - Trigger efficiency
  - PMT gain vs voltage, dark count


One dead channel
ADC Counts (a.u.)
Time (us)
Veto PMT 0
Veto PMT 1
Veto PMT 2
Veto PMT 3
Veto PMT 4
Veto PMT 5
Veto PMT 6
Veto PMT 7

Possible upgrade to DAQ software

1. Current software - saberdaq
    a. Written in C++, object-oriented
    b. Config file - based
    c. Recompilation needed for added features
    d. A little difficult in extending the code to other scenarios

2. I am currently working on another DAQ software project - polaris
    a. **Philosophy : one program for a wide range of DAQ needs.**
    b. Initially only a hobby project for DIY electronics, sensors & DAQ, networking …
    c. Currently used in measuring & recording pressure, temperature, etc. for crystal-related work
    d. Written in C++ again, object-oriented, config-file based
    e. Highly modular, functions loaded at runtime instead of compile time
    f. Low software overhead
    g. No additional library dependency
    h. Flexible and easy to extend and scale up

The reality and challenge of DAQ software

1. Too much variety!
    a. Nature of data, bandwidth, sampling rate, ways to visualize, ...
2. High-dependence on hardware
3. Development cost is often high
4. polaris helps to:
    a. Modularize DAQ jobs - independent and intermix
    b. Reduce DAQ software development cost and time
    c. Promote software reuse

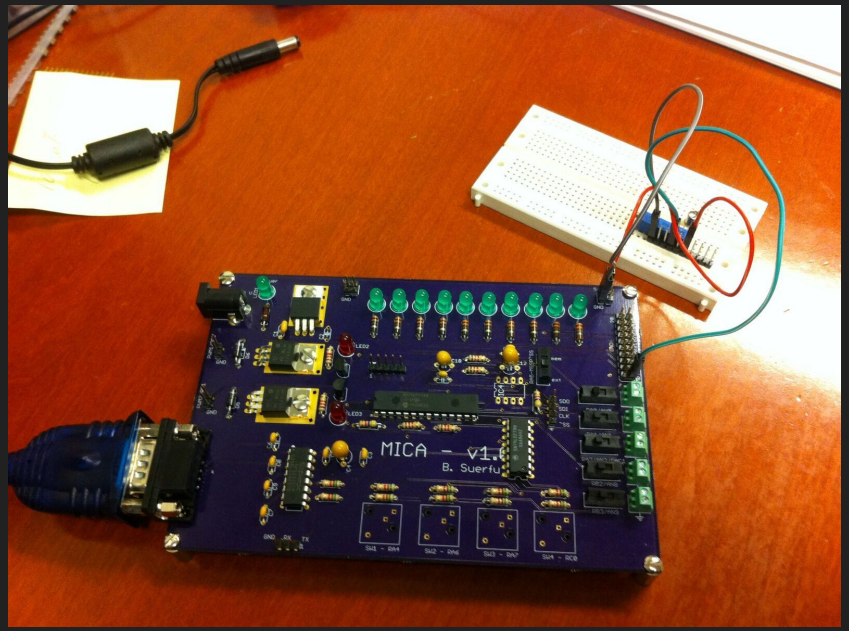
How polaris works

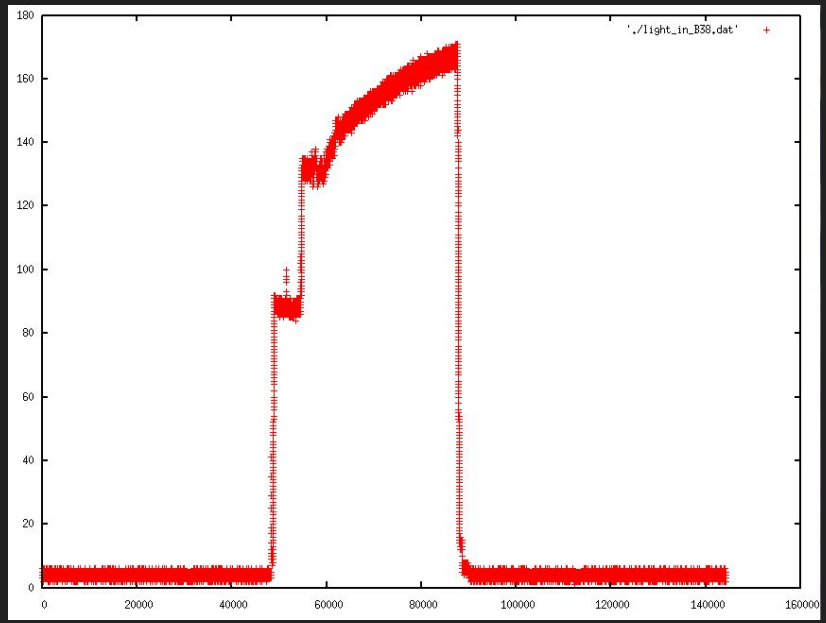
1. In generalizing DAQ three things are unavoidable:
    a. How data is acquired
    b. What does the data look like
    c. How data is stored

2. polaris works by:
    a. User writes libraries specifying how to get data, and how to write them onto disk, and how to visualize them
    b. User specifies in config file where to look for above libraries
    c. polaris will load the needed libraries at runtime and coordinates between different modules throughout the entire DAQ process.

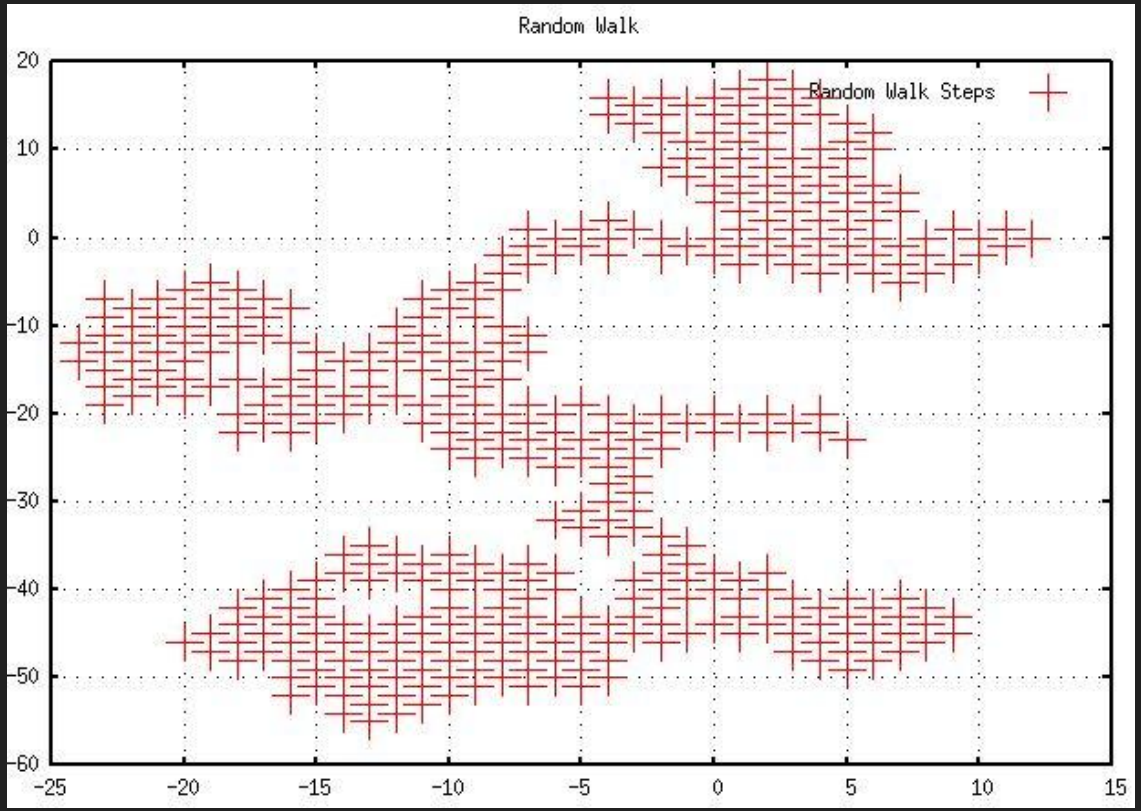
Demo: turning on light in the room

Custom circuit with ADC that reads voltage periodically on a phototransistor





Demo: reading random noise in computer



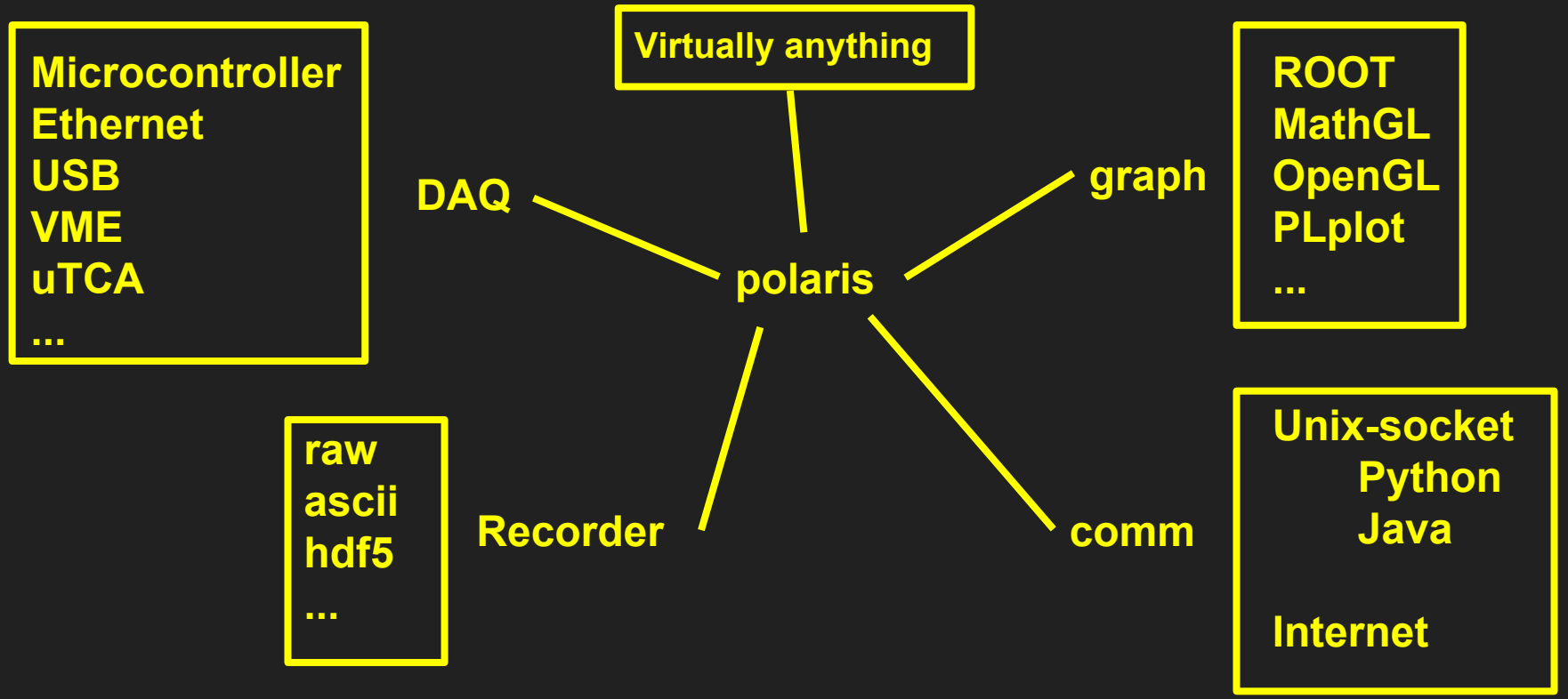
How can polaris improve SABRE DAQ

1. Generalizability:
    a. Using a different hardware, rewrite only the DAQ library and load it at runtime
    b. Need to add a new hardware, write the corresponding plugin and load it at runtime

2. Communication to other programs:
    a. Suppose there are other programs (python, java, labview) that needs communication, simply implement a socket-class and load it at runtime.
    b. Easier integration of slow-control data

3. Visualization:
    a. Want to use Gnuplot, ROOT, Python, plotutil, MathGL, OpenGL, … for visualization, simply write only a new graphics object and load it at runtime.
    b. Different visualization method depending on what is being visualized

How can polaris improve SABRE DAQ



Current status of polaris

1. Can read and record voltage on a phototransistor

2. Reading from V1720
    a. Tested in the absence of PMT signal
    b. Readout no problem

3. Possible field-test in November/December 2017

4. Network interface class currently being implemented
    a. Scalable via master-slave model
    b. Inter-program communication will be achieved

5. uTCA support being added - suggestion from CMS
    a. uTCA - state of the art hardware nowadays

Upgrade to full-scale experiment

1. Trigger:
    a. The underlying digital circuit is block-based, easy to extend to multiple veto / crystal
        i. As easy as adding another IC chip on a breadboard and making connections.
    b. FPGA - OK
    c. Limited pin for ADC trigger input, but on the order ~ 100.
        i. Special ribbon cable might be needed
    d. New functionality can also be added, with slight difficulty

2. ADC:
    a. Will need a preamp - totally new PMT base and enclosure
    b. ground -loop : V1720 has implicit ground on the Connector - Crate - Power line
        i. Floating ground - isolation transformer
        ii. Differential input

3. ADC readout:
    a. Bandwidth no problem : f x N ~ 15000 per optical link