IceRay Testing mini-HOWTO

John Kelley, 25 October 2007
jkelley@icecube.wisc.edu

1. What is This?

This document contains instructions for driving the IceRay test software at UH-Manoa. Unless otherwise noted, all software is installed on the hub `idlpc6`. The system should be booted from the RedHat partition, and the user should be logged in as `aura`.

2. Power-up

To power up the system, first turn on all three relevant power supplies — there are two to the left of the PC monitor connected in series which power the DOR card and thus the mainboard, and there is the 100V rack-mount unit on the bench. The order of power-up is not important.

Next, the DOM mainboard itself can be powered on via the command

```
% on 0 1
```

which powers on DOR card 0, wire pair 1. You should see output indicating that DOM "A" is communicating (there is no unterminated "B" DOM on this pair).

2. DOM Mainboard Prep

The next step is to put the mainboard in the correct state (iceboot) for data taking:

```
% iceboot 01a
```

You should see output indicating that the mainboard is in iceboot.

Now, you also need to check that the `dtsx` software is running. This software attaches a network socket at port 5004 to the Linux device file representing the DOM mainboard. This can be done with the following command:

```
% killall dtsx; dtsxall.pl
```

3. Running the Test Script

The proto-DAQ testing system is a Python script which controls the mainboard and the ICRR through a suite of custom iceboot (the DOM mainboard command-line OS) commands. The top-level `IceRay.py` script and supporting code is run from this directory on `idlpc6`:

```
% cd ~/testdaq/iceray/scripts/run-scripts
```

Usage for the script can be viewed by executing without any command-line arguments:

```
% ./IceRay.py
```

The script currently has two run modes: first, it can acquire forced triggers and save the events (including raw LAB3 waveforms) as binary files, as well as decoded headers in a text file; second, it can perform a DAC threshold scan and save the results via the decoded headers (including scaler rates).

3.1 Forced Triggers / Waveform Acquisition

Forced trigger mode with waveform acquisition can be run with the following arguments to `IceRay.py`:

```
% ./IceRay.py –d 010 –F –f <file prefix> -n <events>
```

The flag `'-d 010'` tells the script to connect to the mainboard at card 0, pair 1, DOM A (=0). `'-F'` specifies forced triggers. The `<file prefix>` should be replaced with a string for naming the event binary files (currently one file per event). The files will be named:

```
<prefix>.1.bin
<prefix>.2.bin
...
```

and so forth, up to the number of triggers / events recorded, specified by the final integer argument `<events>`.

For example,

```
% ./IceRay.py –d 010 –F –f testwf -n 100
```

will acquire 100 forced-trigger waveforms, with filenames `testwf.1.bin`, `testwf.2.bin`, etc. The binary file format is described in this document:

http://www.icecube.wisc.edu/~jkelley/iceray/Virtual Addres CMD.ver1.xls

This will change somewhat, as we have not added the mainboard header and ATWD data to the events yet.

The decoded event headers are all *appended* to the file `headers.txt`, along with an event timestamp (from the PC, not from a real time calibration). The columns in the header file are in the same order as the non-waveform data, as described in the document linked above.

3.2 Threshold Scan

The other mode in which IceRay.py can run is a threshold scan, in which it ramps up all ICRR DAC values from 0 to 4095 and records, via the decoded event headers, all scaler rates at each DAC setting.

The syntax for this type of run is as follows. It is recommended to move the old headers file out of the way first in order to have a clean file for this run:

```
% mv headers.txt headers.txt.bak
% ./IceRay.py –d 010 –F –S -h -n 100
```

The `'-S'` flag switches on the scan. The `'-h'` puts the ICRR in housekeeping mode, in which no waveform data is transferred to the mainboard, only the headers. This saves some time. The `'-n 100'` normally would select how many steps to perform in the scan, but this is currently hard-coded to switch to finer binning in the DAC region of interest. We may fix this at some point.

4. Analysis

Analysis / visualization tools are quite primitive at this stage: a couple of ROOT and gnuplot scripts with basic binary and header file parsing. If you'd like to use either of these, they can be found at

http://www.icecube.wisc.edu/~jkelley/iceray/iceray.html

I can provide usage information upon request.

5. Power-down

To power the system down, first power off the mainboard:

```
% off all
```

Then turn off all three power supplies.

6. Troubleshooting

If the Python script hangs immediately at startup, interrupt it with CTRL-C, and then run it again. Please contact me (jkelley@icecube.wisc.edu) if you have any questions regarding these procedures or run into any unexpected trouble. Thanks!