Below is a list of facts collected in the process of reviewing the microboone project source code.

1. The project source code has 47 “c” files and 26 “h” files containing 6800 and 1700 lines of code, respectively.

2. The source code is structures as follows
   code/
   share/
   src/--- shared daq code
   host/
   assembler/ -- directory has 23 files, 9 of which are applications
   rc/ -- directory has 2 files, 2 of which are applications
   datasim/ -- directory has 3 files, 3 of which are applications

3. Application list

   1) host/assembler/src/**assemblerMain.c**
   arguments: –f crateFile –o outFile
   ->crateFile name format : hardware, system, port, number of boards, total channels
   description: assemblerMain is a receiver/client that receives event data from data from multiple (??) SEB senders.

   2) host/assembler/src/**daqLogd.c**
   arguments: takes no arguments
   description: daqLogd is a network message logger. Received messages are written to a file.

   3) host/assembler/src/**daqLogMain.c**
   arguments: <”line”> (character string)
   description: daqLogMain sends a character string message to daqLogd, which listens on a hardcoded server/port.

   4) host/assembler/src/**daqLogTester.c**
   arguments: <”line”> (character string)
   description: daqLogTester infinitely sends the “line” string to daqLogd.

   5) host/assembler/src/**fake_assembler.c**
   arguments: -f outFile –p port –m host [default =”localhost” ] –e events –v (verbose)
   description: fake_assembler is a receiver/client that receives event data from data from a single SEB senders.

6) host/assembler/src/**fake_assemblerMulti.c**
arguments: -f outFile –p port –m host[="localhost" ] –T SEBcount (number of SEB senders) –e events –v (verbose)
description: fake_assemblerMulti is an SEB receiver/client that receives event data from multiple SEB senders.

7) host/assembler/src/**fake_seb.c**
arguments: -f inputFlie –e numberEventsToSend –p port –m host(="localhost" ] –t sendRepeatCount
description: fake_seb sends fake SEB data to fake_assembler or fake_assemblerMulti.

8) host/assembler/src/**read_data_dump.c**
arguments: -f fileName (unknown whether it is an input or output file name)
description: read_data_dump reads data samples file from a file and dumps them to (???). Data samples have values for channel, offset, q_adc, t_adc.

9) host/datasim/src/**driver.c**
arguments: <nuberOfTimestamps>
description: Process generates data samples as if they were produced by an actual SEB. It calls chunk, digitize, integrate, and MPT functions.

10) host/assembler/src/**read_qt_dump.c**
arguments: -f dataFile
description: read_qt_dump reads data from an input file and prints them to stdout.

11) host/datasim/src/**read.c**
arguments: -f fileName –t tubeNumber (range 1-128) –e eventNuber –c channels
description: reads a file ???

12) host/datasim/src/**test.c**
arguments: takes no arguments
description: A program for testing digitize, integrate and sample functions.

13)host/rc/src/**shmMonitor.c**
arguments: takes no arguments
description: shmMonitor prints a summary generated from the shared memory (shm) segment to stdout.

14) host/rc/src/**shmMonitorFile.c**

arguments: takes no arguments;
description: Process writes an shm summary to a file and possibly prints to stdout.
*concerns: An output buffer is used before it is initialized. The buffer size is a "magic" number.*

15) share/src/**byte_count.c**
arguments: takes no arguments;
description: Process prints sizes of data structures.

16) share/src/**remoteMain.c**
arguments: <list of commands> <remote host>
description: Process sends commands (shell commands ??) to the remoted process, using remote_send() calls.

17) share/src/**remoted.c**
arguments: takes no arguments;
description: Process executes remote commands sent from remoteMain.
*concerns: Potential security violation.*

4. Comments and suggestions:
    a. move the daqLog code into a separate subproject;
    b. factor out daq data formats into a separate subproject;
    c. split sender/receiver functionality into two separate subprojects;
    d. use a secure solution for executing commands remotely, e.g. rgang.py;
    e. adopt a robust message logger developed for nova;
    f. ??? trigger code suggestion ??

Additional suggestions:

g. implement a shared memory display as a Qt application;
h. improve DAQ monitoring by adding process heartbeats;
i. add an ability to collect, save and publish runtime performance metrics to cluster monitoring tools, such as ganglia;
j. factor out the daq configuration into a separate project and make it persistent in a database;
k. improve debug message logging by replacing printfs with the TRACE library.