LIGO Laboratory / LIGO Scientific Collaboration

LIGO- E1000341 *LIGO* 17 June 2010

HAM-ISI LHO test stand:
software, electronic checks and user guides

Ben Abbott, Eric Allwine, Corey Gray,
Vincent Lhuillier, Fabrice Matichard, Hugh Radkins, Jim Warner

Distribution of this document:
Advanced LIGO Project

This is an internal working note
of the LIGO Laboratory

California Institute of Technology
LIGO Project – MS 18-34
1200 E. California Blvd.
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project – NW22-295
185 Albany St
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
P.O. Box 1970
Mail Stop S9-02
Richland WA 99352
Phone 509-372-8106
Fax 509-372-8137

LIGO Livingston Observatory
P.O. Box 940
Livingston, LA 70754
Phone 225-686-3100
Fax 225-686-7189

.

Table of contents:

Introduction ... 3
- Step 1 - Test stand testing ... 3
- Step 2 - Inspection prior starting using the test stand ... 7
- Step 3 - Software check ... 8
- Major lessons learned and actions from CDS ... 15

Appendix 1: Test Stand User Guide ... 16
Appendix 2: Repository User Guide ... 18
Appendix 3: HAM-ISI Front End Simulink Code Instructions ... 19
Appendix 4: Sensors and Actuators Nominal Calibration ... 21

Introduction

In this document, we provide a list of tests to be done to check that the *software and electronics* are ready for the overall assembly testing.

- ## *Step 1 - Test stand testing*

The test stand electronics have been tested after installation. The tests results are summarized in this section and detailed in T1000277-v1.

The tests have to be done only one time per test stand, after installation. They don't have to be redone prior testing each unit. Only the inspection described in section k has to be done prior testing a HAM-ISI unit.

Summary of LIGO-T1000277-v1, aLIGO HAM ISI Test Stand test Plan for LHO Installation, Ben Abbott:

- **Introduction**

The tests described below are required to verify the correct installation and operation of the control and monitoring electronics for an aLIGO Single-stage Internal Seismic Isolation (ISI) electronics Test Stand system. The tests are designed to test the electronics needed to control and monitor a single Horizontal Access Module (HAM) ISI Test Stand. This system should consist of :
2) aLIGO HAM ISI Anti-Alias Chassis
1) AdL Anti-Image Chassis
2) ISI Coil Driver Chassis
3) aLIGO HAM ISI Interface Chassis

Coil driver, Anti aliasing chassis, Anti image chassis and interface chassis used for this test are listed below:

Hardware	Ligo reference	S/N
Coil driver	D0902744	S1000266
		S1000267
		S 1000269
Anti Image filter	D070081	S1000250
Anti aliasing filter	D1000269	S1000246
		S1000247
Interface chassis	D1000067	S1000270
		S1000271
		S1000272

S1000267 has been changed during testing

Figure - From top to bottom – Interface chassis – Anti-aliasing filters
Anti-image chassis – Coil Driver

- **Test Equipment**

- Function Generator
- Digital Multimeter (DMM)
- STS-2/L4C/GS-13 Seismometer Emulator Box
- 25-pinD to two 9-pinD vacuum cable, or in-air cable with the same pinout.
- 25-pin DSub breakout boards
- Voltage Calibrator

- **Input Power**

Turn on the power switches on the rear panel of the Anti-Alias Chassis, Anti-Image Chassis, ISI Coil Driver Chassis, ISI Interface Chassis, and the STS-2 Interface Chassis. Record the current being drawn by the system. Nominal is 330 mA (+18), 310mA (-18), 2.3A (+24), 1.6A (-24).

+18V _3.3_A -18V_3.1_A

+24V_2.3_A -24V_1.6_A

- **Geophone channel Tests, GS-13s:**

Turn the Seismometer selector knob on the L4C/GS-13 Emulator box to the L4C setting. Plug in the appropriate connector of the two D9 connectors on the in-vac wiring cable. Check to see that the front panel power lights are green. If either is unlit, or red, disconnect the cable, and troubleshoot the power problem. Once the lights are both green, turn the front panel switch to "ON". Turn the front panel oscillator switch to "INT". Look at the appropriate channel with dataviewer, and verify that there is a signal. Using a voltage calibrator, or equivalent, put a 3.6VDC "L4C Test Monitor" BNC. Record the readback in counts in the table below.

GS-13 Channel	Amplitude (counts) Nominal is 700 (~350 to -350)	Pressure readback Nominal is -23520 for vertical and +21780 for horizontal
H1	700	21777
V1	700	-23529
H2	700	21780
V2	700	-23510
H3	700	21742
V3	700	-23517

- **L4Cs**

Keep the Seismometer selector knob on the L4C/GS-13 Emulator box to the L4C setting. Plug in the appropriate connector of the two D9 connectors on the in-vac wiring cable. Check to see that the front panel power lights are green. If either is unlit, or red, disconnect the cable, and troubleshoot the power problem. Once the lights are both green, turn the front panel switch to "ON". Keep the front panel oscillator switch to "INT". Look at the appropriate channel with dataviewer, and verify that there is a signal. Record its value in the table below. Using a voltage calibrator, or equivalent, put a 3.6VDC "L4C Test Monitor" BNC. Record the readback in counts in the table below.

L4C Channel	Amplitude (counts) Nominal is 700 (~350 to -350)	Pressure readback Nominal is -23520 for vertical and +21780 for horizontal
H1	700	21789
V1	700	-23520
H2	700	21765
V2	700	-23510
H3	700	21784
V3	700	-23517

- **Coil Driver Tests**

For each channel, put a 5Ω power resistor into the appropriate output. Put a minus (-)10,000 count DC level (which should correspond to 3.0VDC across the resistor) into the Coil Driver input, and read back the Vmon and Imon signals. Move the resistor to each channel under test, and record the results in the table below.

POD/Channel	Volts across resistor (3.0V Nom.)	Vmon (counts) (3438 Nom.)	Imon (counts) (590 Nom.)	Conversion of counts to Amps (970 Nom)
H1	3.0 V	3436	586	957.5 cts/A
V1	3.0 V	3435	589	981.6 cts/A
H2	3.0 V	3429	588	960.8 cts/A
V2	3.0 V	3438	589	981.7 cts/A
H3	3.0 V	3437	592	967.3 cts/A
V3	3.0 V	3437	580	966.7 cts/A

- **Capacitive Position Sensor Tests**

Put in a 1Vp-p 10 Hz sine wave in on the appropriate pins below. Check the output of the capacitive position sensor in Data viewer. Make sure that the power out on the connector labeled Power Out is +18 at pin 1 and -18 at pin 3 with respect to the middle pin (GND).

Channel	Pins	Output Nom. 3,000 cts. p-p	Voltage at Power Out good?	Pass/Fail
V1	1(+) 14(-)	3,000	Yes	Pass
H1	2(+) 15(-)	3,000		Pass
V2	3(+) 16(-)	3,000		Pass
H2	4(+) 17(-)	3,000	Yes	Pass
V3	5(+) 18(-)	3,000		Pass
H3	6(+) 19(-)	3,000		Pass

▪ *Step 2 - Inspection prior starting using the test stand*

In the previous section we presented the exhaustive tests done after the test stand installation. In this section we simply verify that the test stand electronics are functioning prior starting testing.

What is the test stand:

- Brief description
- Picture
- Link to the documents and wiki page
- D1001412 HAM-ISI System wiring

Test stand test template: T1000277.
Test stand test results: Appendix 1.

Electronics:

- Set electronic alimentation +24V +18V.
- How to set up the current limit with the right hand side knob?
- Grounding of electronics: ADE boards.
- Make sure all the coil drivers fuses LEDS are green
- The coil driver temperature LEDs are off.

Figure - Coil Driver with green LEDs

- ***Step 3 - Software check***

For Detailed info on all software, please see Appendices 4 & 5.

- **ssh Connection From A Workstation To The Front End (i.e. Test Stand Computer)**

The test stand computer configuration is described in appendix 4. An ssh (secure shell) connection is necessary to access and work on the Front End computer.

To test the connection, open a terminal on a workstation and connect using the command:

> *ssh -X controls@stormy.ligo-wa.caltech.edu*
> ***Password: contact Corey Gray***

ssh connection test: **Passed: ___** **Failed: ___**

- **SVN**

The SVN/repository is a version control system to keep track of all of our files. A copy of the repository should be installed directly onto the Front End. It is located here:

/opt/svncommon/seisvn/seismic/

The folder for the aLIGO HAM-ISI testing is:

/opt/svncommon/seisvn/seismic/HAM-ISI/X1

To test the existence & functionality of the Front End SVN, do the following when ssh-ed into the Front End:

> *cd /opt/svncommon/seisvn/seismic/HAM-ISI/X1*
> *svn update*

One should be able to (1) find this folder and (2) when an update is run, see new files updated and/or the latest revision the SVN is running with.

SVN test: **Passed: ___** **Failed: ___**

- **MEDM**

MEDM is the Graphical User Interface of the Front End. It allows one to access filter banks, channel names, offsets, basis change matrices, the Watchdog, Master Switch, etc. for our system.

Check whether one can open our medm files and if they are functional.

For the Test Stand we are using Mac computers to run our software. In order to use the application medm (as well as Dataviewer and DTT), one must make sure the application X11 is open. Then one must open a terminal and enter the following:

> *ssh –X controls@stormy.ligo-wa.caltech.edu*
> *medm*

This will open a small gray window where one can open up medm files (.adl). Our medm files are located at:

/opt/rtcds/geo/g1/medm/g1isiham
&
/opt/rtcds/geo/g1/medm/g1x01

With the small medm gray window (which should be in "Edit" mode), perform a File/Open of the "Overview" medm at:

/opt/rtcds/geo/g1/medm/g1isiham/G1ISIHAM_HAM_OVERVIEW.adl

Make sure the Overview window looks alright (click to open sub-windows and make sure they look normal and are displaying data).

MEDM test: **Passed: ___** **Failed: ___**

- **Dataviewer**

Dataviewer is an application for monitoring signals in real-time and as trends. It enables one to visualize Epics and DAQ channels.

Test whether you have Dataviewer by opening it up and looking at data. Open a terminal and enter the following:

> *ssh –X controls@stormy.ligo-wa.caltech.edu*
> *dv*

This will open the Dataviewer interface and the terminal will display running diagnostics for Dataviewer. Under the Signal tab, select a few channels (at G1:ISI-HAM). Then on the Realtime tab, Start a session and check to see if one can see the selected channel's signal displayed in realtime. Additionally, one should also look at trend data under the Playback tab (make sure to pick a channel that ends with "_DAQ").

Dataviewer: **Passed: ___** **Failed: ___**

- **DTT**

DTT (Diagnostic Test Tools) is a GUI mainly used to measure power spectrums and transfer functions.

Test functionality of DTT by opening it up and looking at data. Open a terminal and enter the following:

> *> ssh –X controls@stormy.ligo-wa.caltech.edu*
> *> diaggui*

This will open the DTT GUI. Select an ISI Test Stand channel, and Start a Power Spectrum. Also, run a transfer function and confirm whether one can run an excitation via DTT.

DTT: **Passed: ___** **Failed: ___**

- **Awgstream**

Awgstream is the tool used to inject excitations for the test stand system. The command for setting up an excitation via awgstream is this:

In Terminal:

> awgstream <excitation channel name> <data rate> <waveform file> <scale> <gps time>

In Matlab:

> system('awgstream <excitation channel name> <data rate> <waveform file> <scale> <gps time>')

To test awgstream in a terminal, pick an excitation channel (also watch it on Dataviewer), and make sure your OUTPUTs are all OFF. Open a terminal and enter the following (as an example):

> ssh –X controls@stormy.ligo-wa.caltech.edu
> awgstream G1:ISI-HAM_DAMP_V1_EXC 2048 /tmp/isi/sch_signal_temp.txt 1 961354200 &

To test awgstream in matlab, enter this at the command line:

> system('awgstream G1:ISI-HAM_DAMP_V1_EXC 2048 /tmp/isi/sch_signal_temp.txt 1 961354200')

If you see a sensible excitation occur during both of these tests, awgstream is installed. As for its true functionality, one will have to run real transfer functions and monitor results accordingly to confirm whether awgstream is working properly.

Awgstream: **Passed: ___** **Failed: ___**

- **Foton Filter Coefficient File (.txt)**

The Filter File contains all of the filters for the g1isiham filter banks. The only filters we'll probably use will be for the Damping Path, Blend Path, and possibly Isolation Path. For many of the early Post-Assembly Tests, we run without any filters ON, so this file isn't as important, but once you want to run with any of these filters you will want to make sure this file exists and is ready to go.

To check for the Filter File, see if the file is in place:

> *ssh –X controls@stormy.ligo-wa.caltech.edu*
> *cd /opt/rtcds/geo/g1/chans/G1ISIHAM.txt*

One should also check filter banks via medm. If you have filters designed and installed, make sure they are loaded into the appropriate filter banks (their names occupy filter module locations).

Foton Filter Coefficient File: Passed: ___ Failed: ___

- **Matlab**

Matlab is the main tool used to work with the ISI model and only tool to run batches of transfer functions.

To check whether you have Matlab installed for the Test stand, one should simply try running it:

> *ssh –X controls@stormy.ligo-wa.caltech.edu*
> *matlab*

Matlab: Passed: ___ Failed: ___

- **mDV & ligodv**

mDV is used to import data (DAQ channels) to Matlab through the NDS via the Matlab function get_data & ligodv is basically the same as mDV except it includes a graphical user interface.

To check mDV & ligodv, make sure suite of files are installed on the frontend:

> *ssh –X controls@stormy.ligo-wa.caltech.edu*
> *cd /opt/apps*
> *ls*

Here you will want to make sure you have ligodv & mDV folders. Inside these folders are various files; there should be various ".m" files in here. An important file in mDV is "get_data.m".

Test mDV, by opening Matlab (as noted above) on the frontend and using “get_data” on a test stand channel. As a test, try something like the following:

> *drivedata=get_data('G1:ISI-HAM_GEOPF_H1_IN1_DAQ', 'raw', 962446694, 2)*
> *plot_struct(drivedata)*

This should produce a plot of data from your geophone.

mDV & ligodv: **Passed: ___** **Failed: ___**

- **Epics bin area tools: namely tconvert, caget/caput commands, gpsclock, etc.**

There are an array of CDS tools one will need on the test stand. For instance, tconvert is a tool which gives you gps times, and is used in all of our Matlab transfer function scripts. One will want to make sure we have these useful tools on the test stand frontend.

Check to see if these tools are installed on the frontend:

> *ssh –X controls@stormy.ligo-wa.caltech.edu*
> *cd /opt/apps/bin*

Here you should find a long list of files. At the very least, make sure tconvert is here. You can also try out tconvert:

> *tconvert now*
> *962449432*

epics bin area Tools: **Passed: ___** **Failed: ___**

- **.ini file**

The ini file defines the channels that will be recorded by the DAQ. To add channels to the DAQ, you have to secured shell to the front end and go to the chans/daq directory:

> *ssh -X controls@stormy.ligo-wa.caltech.edu*
> *cd /opt/rtcds/geo/g1/chans/daq/*

Before making changes, save a copy of the current file in the /archive directory.

Open the 'G1ISIHAM.ini' file via emacs or some other editor:

> *emacs G1ISIHAM.ini*

The channels you need have to be un-commented in the .ini file, and the '*acquire*' argument value changed to 1.

Example:

#G1:ISI-HAM_DAMP_V3_EXC_DAQ
#acquire=0
#chnnum=20042
#datarate=2048
#datatype=4

Modify to:

G1:ISI-HAM_DAMP_V3_EXC_DAQ
acquire=1
chnnum=20042
datarate=2048
datatype=4

You can also change the sample frequency but 2048 Hz is advised for simplicity (even for low frequencies measurements)

Once the .ini file has been modified and saved, we have to Reload the DAQ (by pressing "DAQ Reload") on the medm screen titled 'G1ISIHAM_GDS_TP_CUSTOM.adl' and Restart the FrameBuilder (This is accessed from the red pull-down button on this medm).

After you verify the Chan Count increased by the number of channels you un-commented (look at the Chan Count value on the medm noted above), telnet into the local host and enter the 'shutdown' command:

> *telnet localhost 8087*
> *shutdown*

This last action reboots the FrameBuilder and the 'UpTime' counter will restart at 0.

When the FrameBuilder comes back, one can check the list of all the Test Stand channels by entering the following:

> *chndump*

This command lists all the Test Stand channels and provides other information (such as samping rate and whether they are recorded by the DAQ).

- **Dataviewer**

Data viewer is a real time signal monitoring. It enables to visualize and get statistics on Epics and DAQ channels.

- **DTT**

DTT (Diagnostic tools) is used mainly used to measure power spectrums and transfer functions. When several transfer functions has to be measured, batch files executed from Matlab are preferred.

- **.ini file**

The ini file defines the channels that will be recorded by the DAQ. To add channels to the DAQ, you have to secure shell to the front end and go to the following directory :
→ cd /opt/rtcds/geo/g1/chans/daq/

Before making changes, save a copy of the current file in the /archive directory.

open the 'G1ISIHAM.ini' file via emacs or some other editor:
→ emacs G1ISIHAM.ini

The channels you need have to be un-commented in the .ini file and the 'acquire' argument value changed to 1. After you verify the Chan count increased by the number of channels you un-commented, telnet into the local host and enter the 'shutdown' command:
→ telnet localhost 8087
→ shutdown

This last action reboots the FrameBuilder and the 'UpTime' counter will restart at 0.

Channel status changes can be checked in Dataviewer. DAQ channels appear as “float”.

- Open GDS_TP medm screen and make sure everything looks normal there, namely:

- Confirm the front-end is running, the GPS time is changing and it is correct and the blue-green light is flashing.
- Confirm the frame builder is running (uptime is counting up)
- Confirm the DAQ status ix 0x0
- Confirm the timing source is green and is set to TDS
- Make sure the CPU time is below the cycle time (488 usec), otherwise the
front-end code's running too long.

- ***Major lessons learned and actions from CDS***

1) Installation using the SVN checkout and then compile process to make an installation is complex and requires too much expertise.

2) We need to get a better handle on all the 'add on' packages, such as mDv, GDS, ligoDV, awgstream, etc. ie parts we were not involved in writing the code for. Just pulling the code from wherever doesn't mean it will work, as they typically involve various configuration setups, from editing their particular config files to making sure proper Linux shared object files exist.

3) While we worked on setting up a standard installation and directory structure for the CDS real-time and DAQ code, we need to expand that scope to include all additional packages.

4) Focus of the DAQ test systems has been on our real-time and DAQ code. We need to bring in all the other stuff as well.

Actions:

1) Keith is working on putting our release code into what are called RPMs. Essentially, this is just a central repository set up to allow updates and downloads from standard Linux commands, such as yum update. This is somewhat equivalent to doing a Windows update. The GDS tools are in such a repository and Keith has been working with Zweizig on getting it to work (now working and we can build GDS tools again).

The idea here is that, via a simple command like 'yum update CDS', all the latest releases of CDS code source, plus tested executables, would be downloaded to the requesting computer. All necessary configuration information would be passed as well such that environments are automatically setup to run the code.

2) Gather up the latest copies of add-on packages and place them in an RPM as well. It's not always clear, at least to me, where to find these things and/or if they are the latest builds.

3) Finish off our standard install document to include all add on packages and where they are to be installed.

4) Have some CDS folks work with those operating the SEI test systems to gain some expertise as to how their various Matlab scripts work. It would be good to pull off some simple segments to add to our DAQ test system to run as part of our testing of new code releases. I'll have to go back and look, but I vaguely remember that Brian already sent me an example.

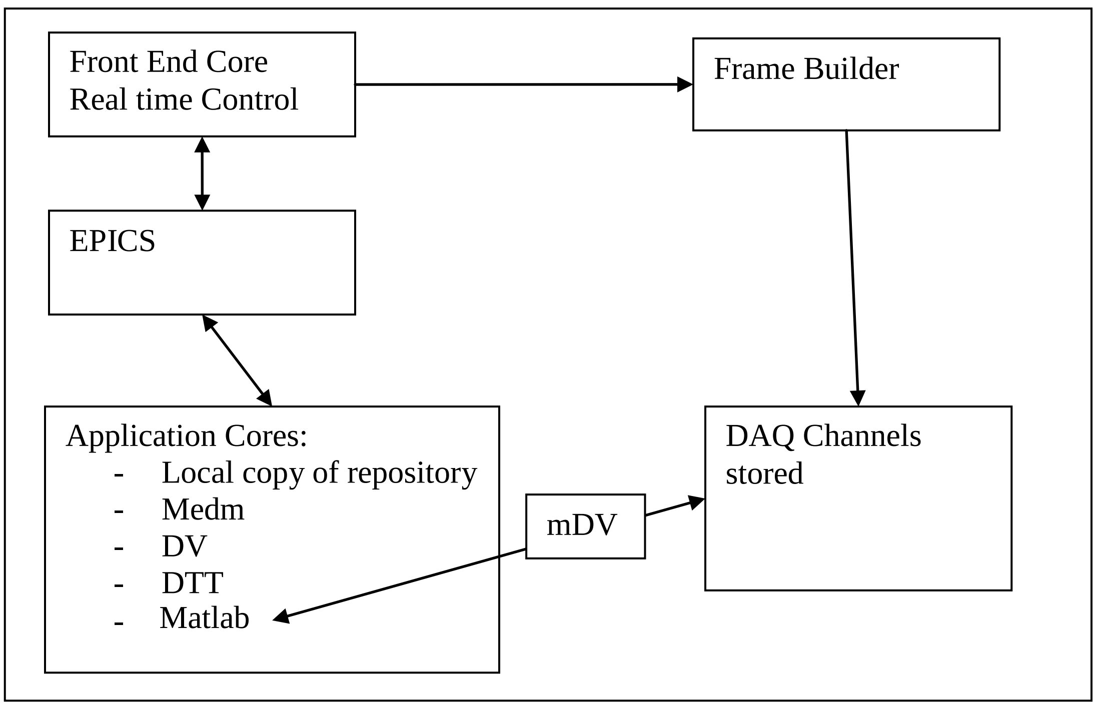
Appendix 1: Test Stand User Guide

http://lhocds.ligo-wa.caltech.edu:8000/advligo/SeiTestStand

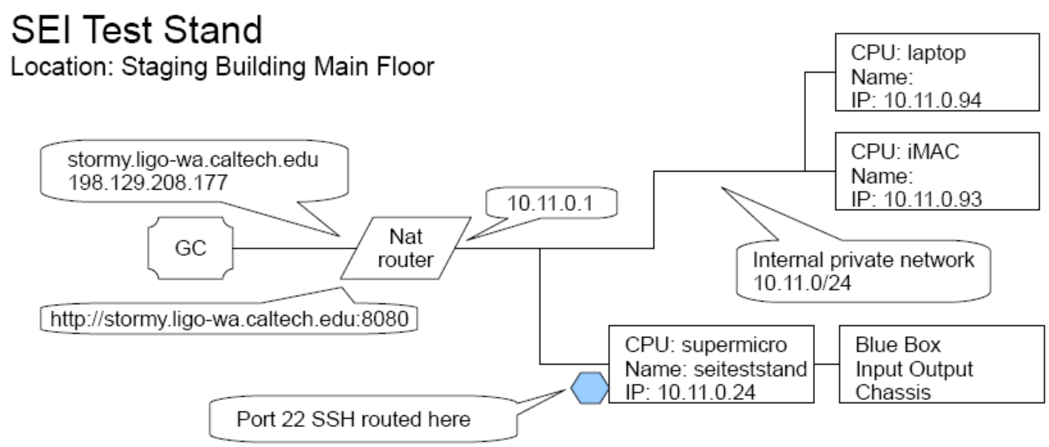
Front end and Software:

One thing that may be useful to be highlighted: one should not be surprised that applications (DTT, svn repository…) were working directly on the front end and not from the workstation like in the sites control rooms. This is the way the test stand works: the front end has a core for the real time computing and other cores for other applications. We are currently using ssh connections from the imac workstation to run applications (i.e matlab, DTT) on the front end.

Seiteststand computer:



Teststand computer connections with other computers and the blue BOX (ADC/DAC):



- **How to connect to the front end.**

Two modes? Question for Dave.

Two ways:

1. From the internal private network??

2. from the GC using secure shell:

Run X11
ssh –X controls@stormy.ligo-wa.caltech.edu
password: call Dave or Corey.

We can also ask David to set up a screen, keyboard and mouse to work directly on the front end.

- **The main path to applications on the front end**

cd /opt/rtcds/geo/g1

Appendix 2: Repository User Guide

1) The repository has been checked out directly on the front end.

It is at:

/opt/svncommon/seisvn/seismic/

2) To connect to the FE and work on the repository

Go to the frontend ("seiteststand") by secure shelling in:

> ssh -X controls@stormy.ligo-wa.caltech.edu

Go to the LHO Test Stand repository folder:

> cd /opt/svncommon/seisvn/seismic

3) Update

4) Add a folder or a file

> svn add <filename>

5) Commit

To make it viewable to everyone else's repository, one must COMMIT their work:

> svn commit --username corey g1isiham.mdl

This opens a VI editor in your terminal to allow you to add a text note to your "file commit". Here are some basic VI commands:

I -- Allows you to insert text
: -- This is for when you want to enter a VI command
w -- Writes your edit
q -- Quits editing session

Once you quit from your VI session, the file is committed to the repository and viewable by others (given that they ran an "svn update").

One would now move this new version of the model over to the build folder:

> cp g1isiham.mdl /opt/rtcds/geo/g1/core/advLigoRTS/trunk/src/epics/simLink

Appendix 3: HAM-ISI Front End Simulink Code Instructions

- Simulink file

The Simulink file location on the repository:

/opt/svncommon/seisvn/seismic/HAM-ISI/X1/Simulink/The Simulink file: g1isiham.mdl

Attention the file name has changed several time (xo1,g1x01, hamisi…) make sure you use g1isiham.mdl.

Once the Simulink file has been modified in the repositary folder it must be copied to the front end build folder where it will be compiled.

The simulink build location on the front end:

opt/rtcds/geo/g1/core/advLigoRTS/trunk/src/epics/simLink/g1isiham.mdl

Editting And Compiling The Test Stand Model (i.e. g1isiham.mdl)

Go to the frontend ("seiteststand") by secure shelling in:

\> ssh -X controls@stormy.ligo-wa.caltech.edu

Go to the LHO Test Stand repository folder:

\> cd /opt/svncommon/seisvn/seismic/HAM-ISI/X1/Simulink

Once here, one can make edits to the model. When making a new file, one will have to ADD it to the SVN.

\> svn add g1isiham.mdl

To make it viewable to everyone else's repository, one must COMMIT their work:

\> svn commit --username corey g1isiham.mdl

This opens a VI editor in your terminal to allow you to add a text note to your "file commit". Here are some basic VI commands:
I -- Allows you to insert text
: -- This is for when you want to enter a VI command
w -- Writes your edit
q -- Quits editing session

Once you quit from your VI session, the file is committed to the repository and viewable by others (given that they ran an "svn update").

One would now move this new version of the model over to the build folder:
> cp g1isiham.mdl /opt/rtcds/geo/g1/core/advLigoRTS/trunk/src/epics/simLink

- How to compile and install

Code Compilation and Installation

The software may be compiled in any user area that includes the cds/advLigo source code tree from the
CDS CVS software repository. This space must be mounted to a computer which has RT Linux installed,
as all compilation must be done on a real-time computer.

To compile the code:

1) Place the MATLAB .mdl file in the directory advLigo/src/epics/simLink

2) Move to the the advLigo directory.

3) Type 'make <sys>', where <sys> is the three letter name of the .mdl file. This command will result in the compilation of all the code, including EPICS.

Once the code is compiled, a few more commands need to be run from the advLigo directory to install the
code for execution.

1) make install-<sys> : This command installs the code in the appropriate directories for execution and makes the automated start-up commands. The EPICS code will be copied to the /cvs/cds/<site>/target/<ifo><sys>epics directory and the front end code will be moved to the /cvs/cds/<site>/target/<ifo><sys> directory.

2) make install-daq-<sys> : This command creates the data acquisition file in the /cvs/cds/<site>/chans/daq directory.

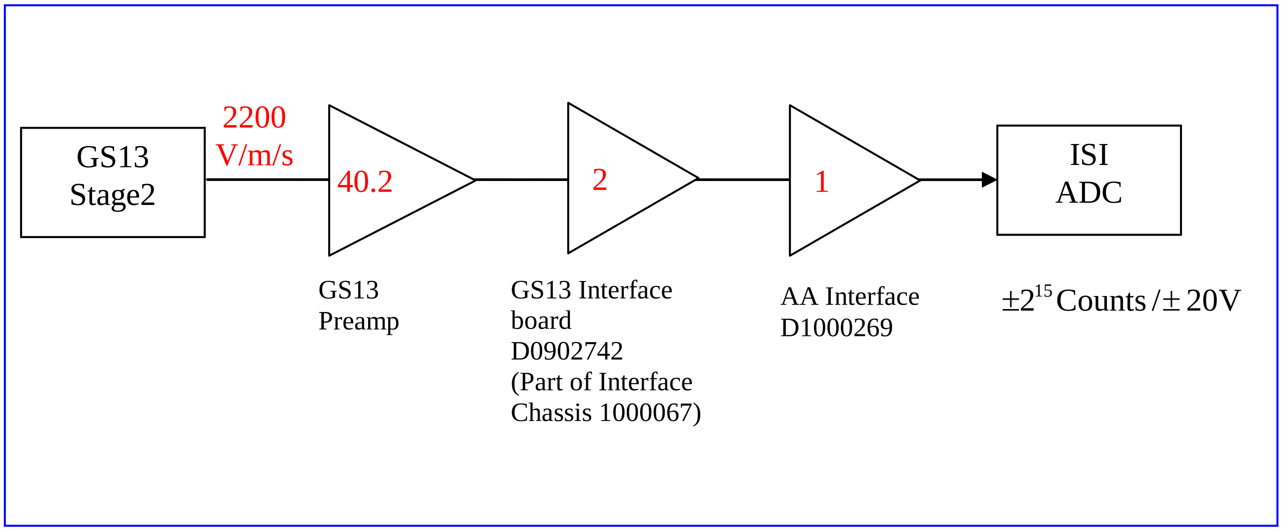
3) make install-screens-<sys> : Installs automatically generated MEDM screens in the /cvs/cds/<site>/medm/<ifo>/<sys> directory.

- Message showing it's well compiled and installed

- Burt restore in Diagnostic.

Appendix 4: Sensors and Actuators Nominal Calibration

GS13 channels Calibration



Frequency dependence:

GS13 Seismometer:

- 3 zeros @ DC
- 2 poles @ 1 Hz

GS13 Interface board

- 1 zero @ 10 Hz
- 1 pole @ 50 Hz, 1 pole @ 2KHz, 1 pole @ 2.24KHz

Anti-Aliasing Chassis D070081

- 3rd order Butterworth @ 10 kHz
- 1 notch @ 2^{16} Hz

CPS Calibration