

# LIGO Laboratory / LIGO Scientific Collaboration

LIGO-T1000561-v2

LIGO

April 3, 2013

# aLIGO CDS

# Software Test Plan

R. Bork

Distribution of this document: LIGO Scientific Collaboration

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

LIGO Hanford Observatory P.O. Box 1970 Mail Stop S9-02 Richland WA 99352 Phone 509-372-8106 Fax 509-372-8137 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 Livingston Observatory P.O. Box 940 Livingston, LA 70754 Phone 225-686-3100 Fax 225-686-7189

http://www.ligo.caltech.edu/

## **Table of Contents**

1	Introduction	3
2	Scope	3
3	Software Flow	3
4	Test System Equipment4.1Stand-alone System4.2Production Test Systems	3
5	Software Builds	5
6	Test Procedures	6
U	6.1 General Code Categories	
	6.2 Test of Built-In Diagnostics	
	6.3 System Tolerance to Hardware Faults	
7	Real-time Code Generator (RCG)	
8	Real-time Control Software	
	8.1 Timing	
	8.1.1 IOP	9
	8.1.2 User Application	9
	8.2 IOC	9
	8.2.1 ADC/DAC tests	
	8.2.2 Binary I/O tests	
	8.3 Code Functional Tests	
	8.4 Real-time Inter-Process Communications (IPC)	
	8.5 Real-time Data Acquisition	
	8.6 DAQ Data Transmission	
	8.7 Arbitrary Waveform Generator / Test Point Manager (awgtpman)	11
9	DAQ System	11
	9.1 DAQDC Testing	
	9.1.1 Load Testing	
	9.1.2 Timing	
	9.1.3 Data Error Detection	
	9.1.4 EPICS Channels	
	9.1.5 Data broadcast	
	9.2 Data Storage	
	9.3 Data Distribution	13
1	0 Integrated System Tests	13
1	1 Test Procedures	14

# 1 Introduction

The purpose of this document is to describe the overall test plan for aLIGO CDS developed and supported software.

# 2 Scope

The scope of the CDS test plan is primarily limited to software developed and/or maintained by CDS staff. This includes:

- 1) Real-time code generator and its products.
- 2) Real-time core software ie software 'wrapper' included in every aLIGO real-time application, which includes I/O drivers, network drivers, DAQ and GDS capabilities, etc.
- 3) Real-time support software ie awgtpman and EPICS and DAQ network interface to realtime applications.
- 4) DAQ software, including receiving, storing, distributing and viewing data, with the two latter items limited to LIGO control room access only.
- 5) Global Diagnostic System (GDS) software, limited to DTT and foton.

That having been said, the test plan does provide for integrated testing with commonly used LIGO control room tools. This testing will be limited to verification that CDS core code changes have not adversely affected the interface to and use of these other tools. 'Other' tools includes:

- 1) Conlog
- 2) Scripts, of various sorts, which have been used to automate LIGO control processes.
- 3) EPICS extensions used in operations, such as MEDM, BURT, ALH, etc.
- 4) Matlab based ligoDV.

# 3 Software Flow

Software development to production follows the following basic steps:

- 1) Development on single computer platform.
- 2) Initial testing on stand-alone system. Most software features can be tested in this configuration (see section 4). Where the code to be tested requires multiple computers, such as network data communication tests, it is moved directly to the production test system.
- 3) Final testing and integration occurs on the production test system. This system provides a much larger scale of computers and networking.

# 4 Test System Equipment

### 4.1 Stand-alone System

A stand-alone system consists of a single CDS computer connected to a single I/O Chassis (IOC). It is self-contained, in that is capable of running all CDS software, including real-time, DAQ and

operator support software. This is the configuration typically deployed to support aLIGO subsystem testing.

## 4.2 Production Test Systems

Two distributed control test systems are in place, one at LHO and one at LLO. These systems contain a subset of the equipment used for IFO control and data acquisition, configured in the same manner. It is intended that, after initial code modules are separately tested, code be moved to these systems for integrated testing.

Equipment included in these systems are:

- Computers
  - Real-time control
  - Boot/Build machines
  - o EPICS gateway
  - Data Acquisition, along with disk storage systems
  - Operator workstations
- I/O Chassis

\_

- Various configurations of supported I/O modules
- Networking equipment
  - EPICS data network
  - DAQ network
  - Real-time deterministic networks
- Test electronics
  - Loopback chassis for signal testing
  - o AA/AI chassis for signal injection/output

#### *LIGO*

# 5 Software Builds

Software builds take place daily on the aLIGO CDS test systems. The Jenkins continuous integration tool set is used to control and record this process (sample web page below). Daily, at midnight, the Jenkins tool automatically:

- 1) Downloads the latest code from the CDS SVN repository. This includes both the CDS core software and the aLIGO user applications.
- 2) Compiles and installs all applications against this latest SVN revision.

Information on the build process and success/failure information is maintained in the Jenkins logs.

Jenkins					Search	? log in
Jenkins LLO						ENABLE AUTO REFRESH
Build History	All L	HO-H1	LLO userapps			
	S	w	Name ↓	Last Success	Last Failure	Last Duration
Build Queue No builds in the queue.		*	make-l1iop	1 day 2 hr ( <u>#15</u> )	4 days 4 hr ( <u>#10</u> )	1 min 28 sec
Build Executor Status # Status		ő,	make-l1oaf	1 day 1 hr ( <u>#5</u> )	1 day 1 hr ( <u>#4</u> )	9 sec
1 Idle			make-l1pemex	1 day 1 hr ( <u>#4</u> )	1 day 1 hr ( <u>#3</u> )	6.7 sec
		(B)	make-l1pemey	1 day 2 hr ( <u>#4</u> )	2 days 19 hr ( <u>#2</u> )	7.9 sec
		*	make-l1psl	1 day 1 hr ( <u>#7</u> )	7 days 3 hr ( <u>#1</u> )	26 sec
		*	make-l1seib1	1 day 1 hr ( <u>#2</u> )	N/A	55 sec
		*	make-l1seib2	1 day 1 hr ( <u>#2</u> )	N/A	52 sec
		*	make-l1seih16	1 day 1 hr ( <u>#2</u> )	N/A	36 sec
		*	make-l1seih23	1 day 1 hr ( <u>#2</u> )	N/A	58 sec
		*	make-l1seih45	1 day 1 hr ( <u>#2</u> )	N/A	57 sec
Brit		<i></i>	make-l1susb123	N/A	2 days 19 hr ( <u>#1</u> )	6.6 sec
2 2 1-2		*	make-l1sush2a	1 day 1 hr ( <u>#2</u> )	N/A	45 sec
		*	make-l1sush2b	1 day 1 hr ( <u>#2</u> )	N/A	17 sec
		*	make-l1sush34	1 day 1 hr ( <u>#2</u> )	N/A	34 sec
		*	make-l1sush34aux	1 day 1 hr ( <u>#2</u> )	N/A	9 sec
		(A)	test-l1seiham23	4 days 2 hr ( <u>#4</u> )	4 days 2 hr ( <u>#3</u> )	20 sec
	Icon:	<u>s M</u> L		Legend SRSS for	all SRSS for failures	RSS for just latest builds

## 6 Test Procedures

Standard CDS software test procedures are being developed to cover testing in both stand-alone and production test facility configurations. For code releases through version 2.5, testing was performed manually for group staff (see LIGO-T1200393 and LIGO-T1200394). For new code releases, test scripts are to be used to automate this process. The Jenkins tool is used to initiate these tests on CDS test systems and record the test results. An example Jenkins screen and test report are shown in the figures below.

Dashboard [Jenkins]								
မာ 🛄 🏭 Apple	Yahoo!	Google I	Maps YouTube	Wikipedia News	(3,413) • Popular •			
Jenkins					💁 search		0	
Jenkins						ENABLE AU	TO REFRESH	
e New Job	AII	+				Made Contract of C	d description	
People	S	w	Name ↓	Last Success	Last Failure	Last Duration		
Build History		*	DiagnosticTest	8 min 34 sec ( <u>#22</u> )	2 days 0 hr ( <u>#15</u> )	2 min 1 sec	$\bigotimes$	
Manage Jenkins Build Queue	0	*	fmc2	1 min 17 sec ( <u>#2</u> )	N/A	1 min 9 sec	$\bigotimes$	
No builds in the queue.	0	*	rcgcore	5 min 0 sec ( <u>#2</u> )	N/A	3 min 9 sec	$\bigotimes$	
Build Executor Status # Status		-	test1	2 days 1 hr ( <u>#15</u> )	1 day 6 hr ( <u>#18</u> )	59 ms	$\bigotimes$	
1 Idle 2 Idle	Icon	<u>s</u> <u>M</u> L	<u>Lege</u>	nd NRSS for all	RSS for failures	S for just lat	<u>est builds</u>	

Help us localize this page

Page generated: Aug 2, 2012 3:53:02 PM Jenkins ver. 1.471

//.

#### *LIGO*

#### LIGO-T1000561-v2

Propert         #59           Reconstruction could unstruct could make an an additional additionadditionadditinadditional additionadditinadditional additinadditin	🕮 🇰 BZintegrati	on Apple Yahoo! Google Maps Yo	ouTube	Wikipedia News (	3,618) -	Popular <b>v</b>			
Note come many name         Nume         Nume </th <th></th> <th>&gt;</th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th>		>							
Title:       10:16:46       Mar. 20 20:33         Testing Wirk weision markets:       314       The Mark Markets:         THE markets:       Mark Markets:       314       The Markets:         Weify 100 I/O Mapping       THE markets:       314       The Markets:         Weify 100 I/O Mapping       THE markets:       314       The Markets:         Weify 100 I/O Mapping       THE markets:       31       The Markets:         Weify 100 I/O Mapping       THE markets:       31       The Markets:         Markets:       The Markets:       31       The Markets:       The Markets:         Markets:       The Markets:       31       The Markets:       The Markets:         Weify 100 I/O Mapping       The Markets:       11       The Markets:       The Markets:         Markets:       The Markets:       The Markets:       The Markets:       The Markets:         Markets:       The Markets:       The Markets:       The Markets:       The Markets:         Markets:       The Markets:       The Markets:       The Markets:       The Markets:         Markets:       The Markets:       The Markets:       The Markets:       The Markets:         Markets:       The Markets:       The Markets:       The Markets:		*****	******	*****	******	********	*****	*****	
THE			ORT *	******	******	*******	*****	******	
The intervention         House the intervention         House the intervention           Verify IOD is running         EDATE 4 3 = 0         14         The intervention           Verify IDD is running         EDATE 4 3 = 0         14         The intervention           Verify IDD is running         EDATE 4 3 = 0         14         The intervention           Verify IDD is running         EDATE 4 3 = 0         14         The intervention           Verify IDD is running         EDATE 4 3 = 0         14         The intervention           ADD Chait Process The intervention         EDATE 4 13         The intervention         PASE           ADD Chait Process The intervention         11 - 11 uses         11 - 11 uses         PASE           ADD Chait Process The intervention         20 +/- 2 uses         20 - 11 PASE         PASE           ADD Chaits Process The intervention         20 -/- 2 uses         20 - 11 PASE         PASE           ADD Chaits Process The intervention         20 -/- 2 uses         -20 - 10 PASE         PASE           ADD Chaits Process The intervention         20 -/- 2 uses         -20 - 10 PASE         PASE           ADD Chaits Process The intervention         20 -/- 2 uses         -20 - 10 PASE         PASE           ADD Chaits Process The interventinte intervention         21 - 2 PASE			14						
Verify D0 is running				************	******	*******	*****	******	
Verify D0 is running									
Verity DD 1/0 Mapping        14        PASS         Total processing time        14        PASS         ADD Data Processing time								PASS/FAIL	
Verify 100 Fishing       TD0       TD0       14       TD0       PASS         Code cycle time       10 succ       14       TD1       PASS         Stock cycle time       10 succ       1       1       TD2       PASS         Stock cycle time       10 succ       1       1       TD2       PASS         ACC botone fining       11-10 usec       11       TD2       PASS         ACC botone fining       11-10 usec       11       TD2       PASS         ACC botone fining       11-10 usec       10       TD2       PASS         ACC botone fining       11-10 usec       10       9995       TD2       PASS         ACC botone fining       11-10 usec       10       9995       TD2       PASS         ACC botone fining       110 4/2 usec       10       9995       TD2       PASS         ACC botone fining       10 4/2 usec       11       TD2       PASS         DAC botone fining       10 4/2 usec       12       TD2       PASS         DAC botone fining       10 4/2 usec       11       TD2       PASS         Verify Loc fining       10 10 (0       10 10 10       10 10 10       10 10 10         Foree chanel ho		Verify IOP is running		STATE & 3 = 0		14		PASS	
Time Spure Gource        TDS        14        PASS         ADD base Process Time						14		PASS	
ADC Data Process Time <2 PASS Total Process Time <10 usec 6 PASS Data Process Time <10 usec 70 PASS Data		Time Sync Source		TDS		14		PASS	
Total Processing Time				15 +/- 2				PASS	
ADC Ductone Timing									
IRG-8 Timing									
Verify Detection of Clocking Errors ****         Add a 64K clock pulse         ADD Ductoms Timing       10 9/1 - 2 usec       20 99996       70 855         DAD Ductoms Timing       10 9/1 - 2 usec       89 9       71 855         DAT Ductoms Timing       10 9/1 - 2 usec       99 9       71 855         DAT Ductoms Timing       10 9/1 - 2 usec       9       71 855         DAT Ductoms Timing       10 9/1 - 2 usec       27 PASS         DAT Ductoms Timing       10 1 9/1 - 2 usec       27 PASS         Verify Detection of ADC Channel Hopping       Porce channel hop       71 code skit 0       71 code skit 0         Porce to ADD Clock       code skit 0       1       1       71 code skit 0         Verify Detection of ADC Timeout (Less of Clocks)       Porce bad ADM Clock       71 code skit 1       71 code skit 1         Porce bad ADM Clock       000-07 code skit 1       1       71 code skit 1       71 code skit 1         Verify Diare Sync to IOP       000-07 code skit 1       71 code skit 1       71 code skit 1       72 code skit 1         Verify Oclostines       000-07 code skit 1       10 code skit 1       72 code skit 1       72 code skit 1       72 code skit 1         Verify Nodel       10 code skit 1       10 code skit 1       74 code skit 1 <td></td> <td>IRIG-B Timing</td> <td></td> <td>11-13 usec</td> <td></td> <td></td> <td></td> <td>PASS</td> <td></td>		IRIG-B Timing		11-13 usec				PASS	
Add a 64% clock pulse       20 +/-2 usec       20 -/-2 usec       PASS         DAC Ductors Tising		DAC Ductone TIming		69-71 usec		70		PASS	
Add a 64% clock pulse       20 +/-2 usec       20 -/-2 usec       PASS         DAC Ductors Tising		Verify Detection of Clocking Fi	rrors *	***					
<pre>IRIC = 5 Timing 0995 - 9999 9995 - 9999 9955 - 9999 PASS DAC Ductors Timing 10 +/. 2 usec 59 PASS BAC Ductors Timing 51-33 usec 52 PASS Verify Detection of ADC Channel Hopping Force channel hop code exit 0 5 PASS Verify Detection of ADC Timeout (loss of clocks) Force band ADC clock code exit 1 1 PASS Verify Detection of ADC Timeout (loss of clocks) Force band ADC clock code exit 1 7 PASS Verify Detection of ADC Timeout (loss of clocks) Force band ADC clock code exit 1 7 PASS Verify Size Sync to IOP IOP (0) 7 PASS Verify Size Sync to IOP IOP (0) 7 PASS Verify for Consections ADC Modules 7 IOP (0) PASS Verify rote Times IOP (0) 7 PASS Verify Consections ADC Module 200 IOP (0) PASS Verify Consections ISK Model 210 210 IOP (0) 8 PASS Check 4 Record CUU Times ISK Model 410 10 IOP (0) 8 PASS Check 4 Record CUU Times ISK Model 410 IOP (0) 8 PASS Check 4 Record ADC-&gt;DAC LoopBack Times ISK Model 122 III PASS Check 4 Record ADC-&gt;DAC LoopBack Times ISK Model 122 III PASS Check 4 Record ADC-&gt;DAC LoopBack Times ISK Model 122 III III PASS Check 4 Record ADC-&gt;DAC LoopBack Times ISK Model III III IIII</pre>		Add a 64K clock pulse							
DAC Ductone Timing									
Skip a 648 clock pulse       -10 4/- 2 usec									
INIC-B Timing		Skip a 64K clock pulse							
DAC Duotone Timing									
Verify Detection of ADC Channel Hopping Force channel hop									
Porce channel hop        code exit 0        5        PASS         Verify Detection of ADC Timeout (loss of clocks) Force bad ADC clock        code exit 1        1        PASS         Start Testing with Slave Processes       Verify Slave Sync to IOP        IOP (8)       8        PASS         Verify I/O Connections        OK-0xf        7        PASS         Verify Cycle Times        OK-0xf        1        PASS         31 Model        0K-0xf        1        PASS         32K Model        0K-0xf        1        PASS         32K Model        27 to 33        1        PASS         2X Model        242 to 248		5110 2000010 11m211g							
<pre>Verify Detection of ADC Timeout (loss of clocks) Force bad ADC clock code exit 1 1 PASS Start Testing with Slave Processes Verify Slave Sync to IOP IOP (8) 8 PASS Number PASS OK-ORT 7 PASS DACC Modules OK-ORT 7 PASS Number PASS Verify Cycle Times 32K Model 5% to 5% to 6% 2% 2% 2% 2%</pre>						_			
Force bad ADC clock        code exit 1        1        PASS         Start Testing with Slave Processes         Verify Slave Sync to IOP        IOP (8)        8        PASS         ADC Modules        0K-7        7        PASS         DACK Modules        0K-7        PASS         Verify Cycle Times        0K-64        62        PASS         Verify Model        58 to 64        62        PASS         XE Model        464 to 492        489        PASS         XE Model        <10		Force channel hop		code exit 0		5		PASS	
Start Testing with Slave Processes         Verify JA Connections         ADC Modules         DAC Model         DAC Times         DAC Model         DAC Times         DAC Model         DAC Model     <		Verify Detection of ADC Timeout	t (loss	of clocks)					
Verify Slave Sync to IOP       IOP (8)        8        PASS         NCC Modules        OK=7        7        PASS         Verify Cycle Times        OK=0xf        15        PASS         Verify Cycle Times        27 to 33        11        PASS         J2K Model        27 to 33        12        PASS         ZK Model        242 to 248        62        PASS         ZK Model        464 to 492        459        PASS         ZK Model        <10		Force bad ADC clock		code exit 1		1		PASS	
Verify Slave Sync to IOP       IOP (8)        8        PASS         NCC Modules        OK=7        7        PASS         Verify Cycle Times        OK=0xf        15        PASS         Verify Cycle Times        27 to 33        11        PASS         J2K Model        27 to 33        12        PASS         ZK Model        242 to 248        62        PASS         ZK Model        464 to 492        459        PASS         ZK Model        <10		Start Testing with Slave Proces	8868						
Verify I/O Connections ACC Modules 0K=7 7 PASS Verify Cycle Times 32K Model 0K=0 15 PASS 32K Model 58 to 64 62 PASS 16K Model 58 to 748 62 PASS 2K Model 484 to 492 489 PASS 2K Model 484 to 492 8 PASS 2K Model <10 8 PASS 2K Model <12 13 PASS 18 bit DAC Times 488 488 PASS 4K Model 122 122 PASS 4K Model 122 122 PASS 4K Model 122 76 PASS 4K Model 76 PASS 4K Model 122 76 PASS 4K Model 123 76 PASS 4K Model 126 76 PASS 4K Model 127 PASS 4K Model 128 76 PASS 2K Model 120 PASS 4K Model 120 PASS 4K Model 120 PASS 4K Model 120 PASS 4K Model 10 PASS 10A TEST COMPLETE - NO ERRORS Proform Basic Memory Checks Verify No TP Set 0 PASS DIAC TEST COMPLETE - NO ERRORS PIAC TEST COMPLETE - NO ERRORS PIAC TEST COMPLETE - NO ERRORS PIAC TEST COMPLETE - NO ERRORS									
ADC Modules          OK-7          7          PASS           Verify Cycle Times          0K-0xf          11          PASS           15K Model          55 to 64          62          PASS           15K Model          242 to 248          62          PASS           16K Model          242 to 249          245          PASS           2K Model          464 to 492          245          PASS           2K Model          <10				IOP (8)		8		PASS	
DACC Modules 0K=0xf 15 PASS Verify Cycle Times 32K Model 27 to 33 31 PASS 16K Model 242 to 248 62 PASS 2K Model 242 to 248 489 PASS 2K Model 484 to 492 489 PASS Check & Record PUTimes 32K Model < 10 3 PASS 2K Model < 16 8 PASS 2K Model < 16 8 PASS 2K Model < 16 PASS 2K Model < 20 13 PASS Check & Record ADC->DAC LoopBack Times - 18 Bit DAC Times 10P Model 61 61 PASS 10F Model 198 PASS 4K Model 198 PASS 4K Model 198 PASS - 16 Bit DAC Times - 176 PASS BASS 				08=7		7		DAGG	
121K Model        27 to 33        31        PASS         16K Model        55 to 64        62        PASS         2K Model        484 to 492        489        PASS         2K Model        484 to 492        489        PASS         Check & Record CPU Times        <10									
16K Model        58 to 64        62        PASS         2K Model        242 to 248        245        PASS         2K Model        484 to 492        489        PASS         2K Model        <10									
4K Model        242 to 248        245        PASS         2K Model        489 to 492        489        PASS         Check & Record CPU Times       3        98        PASS         3K Model        <10									
Check & Record CPU Times 32K Model < 10 3 PASS 16K Model < 16 8 PASS 2K Model < 20 13 PASS Check & Record ADC->DAC LoopBack Times - 18 Bit DAC Times - 18 Bit DAC Times - 18 GM Model 61 61 PASS 32K Model 198 PASS 4K Model 198 PASS 4K Model 610 610 PASS 4K Model 610 610 PASS 2K Model 610 76 PASS 2K Model 610 76 PASS 2K Model 625 PASS 4K Model 625 PASS 4K Model 625 PASS 4K Model 625 PASS 4K Model 625 PASS Perform Basic Memory Checks Verify No TP Set 0 PASS Verify No DAC Outputs 0 PASS Porcess leaked file descriptors. See http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build for									
32K Model         0        3        PASS         16K Model         10        8        PASS         2K Model         20        13        PASS         2K Model         20        13        PASS         2K Model           61        PASS         Check & Record ADC->DAC LoopBack Times       -         61        PASS         10P Model             PASS         32K Model <td></td> <td>2K Model</td> <td></td> <td>484 to 492</td> <td></td> <td>489</td> <td></td> <td>PASS</td> <td></td>		2K Model		484 to 492		489		PASS	
32K Model         0        3        PASS         16K Model         10        8        PASS         2K Model         20        13        PASS         2K Model         20        13        PASS         2K Model           61        PASS         Check & Record ADC->DAC LoopBack Times       -         61        PASS         10P Model             PASS         32K Model <td></td> <td>Check &amp; Record CPU Times</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>		Check & Record CPU Times							
4K Model        < 16				< 10		3		PASS	
2K Model        13        PASS         Check & Record ADC->DAC LoopBack Times         - 18 Bit DAC Times         IOP Model        61        PASS         JXK Model        122        PASS         JK Model        198        PASS         JK Model        198        PASS         JK Model        198        PASS         JK Model        198        PASS         JK Model        610        PASS         ZK Model        610        PASS         ZK Model        122        PASS         JCP Model        76        PASS         JCM Model        122        PASS         JCP Model        198        PASS         JCM Model        198        PASS         JCM Model        198        PASS         ZK Model						-			
Check & Record ADC->DAC LoopBack Times         - 18 Bit DAC Times         IOP Model        61        PASS         32K Model        122        PASS         16K Model        198        PASS         2K Model        610        PASS         32K Model        76        PASS         32K Model        122        PASS         32K Model        76        PASS         32K Model        198        PASS         2K Model        198        PASS         2K Model									
- 16 Bit DAC Times IOP Model 61 PASS 32K Model 122 122 PASS 16K Model 198 198 PASS 16K Model 488 488 PASS 2K Model 610 610 PASS 2K Model 76 PASS 10P Model 76 PASS 32K Model 122 122 PASS 16K Model 198 PASS 16K Model 625 625 PASS 2K Model 625 PASS 2K Model 0 PASS Verify No TP Set 0 PASS Verify No TP Set 0 PASS Verify No TP Set 0 PASS Perform Basic Memory Checks Verify No TP Set									
IOP Model        61        PASS         32K Model        122        122          16K Model        198        198          4K Model        488        610        PASS         4K Model        610        610        PASS         -16 Bit DAC Times        610        610        PASS         IOP Model        76        PASS        PASS         16K Model        122        122        PASS         32K Model        122        122        PASS         16K Model        198        198       PASS         4K Model        503        PASS         2K Model        625        PASS         Verify No DAC Outputs        0        PASS         Verify No DAC Outputs        0			ck Time:	В					
32K Model        122        PASS         16K Model        198        PASS         4K Model        488        PASS         2K Model        610        610          - 16 Bit DAC Times        610        610          10P Model        76        PASS         32K Model        122        122          16K Model        198        PASS         16K Model        198        PASS         2K Model        503        PASS         2K Model        625        PASS         2K Model        625        PASS         2K Model        625        PASS         2K Model        0        PASS         2K Model        625        PASS         2K Model        0        PASS <td></td> <td></td> <td></td> <td>61</td> <td></td> <td>61</td> <td></td> <td>PASS</td> <td></td>				61		61		PASS	
4K Model        488        PASS         2K Model        610        PASS         - 16 Bit DAC Times       IOP Model        76        PASS         32K Model        122        122        PASS         16K Model        198        198       PASS         4K Model        503        PASS         2K Model        625        PASS         2K Model	3			122					
2K Model        610        PASS         - 16 Bit DAC Times       IOP Model        76        PASS         32K Model        122        122        PASS         16K Model        198        198        PASS         16K Model        503        PASS       PASS         2K Model        625        PASS         2K Model        625        PASS         Verify No TP Set       0        0          Verify No DAC Outputs       0        0       PASS         DIAG TEST COMPLETE - NO ERRORS       Process leaked file descriptors. See <a href="http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build">http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</a> for									
- 16 Bit DAC Times IOP Model 76 PASS 32K Model 122 PASS 16K Model 198 PASS 16K Model 503 9ASS 4K Model 503 9ASS 2K Model 625 PASS Perform Basic Memory Checks Verify No TP Set 0 PASS Verify No TP Set 0 PASS Verify No DAC Outputs 0 PASS DIAG TEST COMPLETE - NO ERRORS Process leaked file descriptors. See http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build for									
32K Model          122          122          PASS           16K Model          198          198          PASS           4K Model          503          503          PASS           2K Model          625          625          PASS           Verify No TP Set          0          625          PASS           Verify No DAC Outputs          0          PASS           DIAG TEST COMPLETE - NO ERRORS          0          PASS           Process leaked file descriptors. See <a href="http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build">http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</a> for						010			
16K Model        198        PASS         4K Model        503        PASS         2K Model        625        625          Perform Basic Memory Checks       Verify No TP Set        0        PASS         Verify No DAC Outputs        0        PASS         DIAG TEST COMPLETE - NO ERRORS       Process leaked file descriptors. See <a href="http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build">http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</a> for									
4K Model        503        PASS         2K Model        625        625          Perform Basic Memory Checks       Verify No TP Set        0        PASS         Verify No DAC Outputs        0        PASS         DIAG TEST COMPLETE - NO ERRORS       Process leaked file descriptors. See <a href="http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build">http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</a> for									
2K Model        625        PASS         Perform Basic Memory Checks Verify No TP Set        0        PASS         Verify No DAC Outputs        0        PASS         DIAG TEST COMPLETE - NO ERRORS Process leaked file descriptors. See <a href="http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build">http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</a> for									
Verify No TP Set 0 PASS Verify No DAC Outputs 0 PASS DIAG TEST COMPLETE - NO ERRORS Process leaked file descriptors. See <u>http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</u> for									
Verify No TP Set 0 PASS Verify No DAC Outputs 0 PASS DIAG TEST COMPLETE - NO ERRORS Process leaked file descriptors. See <u>http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</u> for		Perform Pasia Menory Chaster							
Verify No DAC Outputs 0 PASS DIAG TEST COMPLETE - NO ERRORS Process leaked file descriptors. See <u>http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</u> for				0		0		PASS	
Process leaked file descriptors. See <u>http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</u> for									
Process leaked file descriptors. See <u>http://wiki.jenkins-ci.org/display/JENKINS/Spawning+processes+from+build</u> for									
	1		s. See 1	http://wiki.jenki	ns-ci.o	rg/display	JENKI	NS/Spawning+processes+fre	om+build for

### 6.1 General Code Categories

The following major sections of this document list the testing to be done by major CDS subsystems, which include:

- 1) Real-time Code Generator (RCG)
- 2) Real-time Executable Software
- 3) Data Acquisition (DAQ) System Software

The final section includes integrated system testing, which includes the above, plus testing with operator interface software and other software tools.

### 6.2 Test of Built-In Diagnostics

Many of the CDS core software components contain various system diagnostics, such as timing and network error detection. Since these diagnostics will be used in various tests, it is important to first verify the proper operation of these diagnostics. Therefore, a single test procedure will be provided to do this verification.

### 6.3 System Tolerance to Hardware Faults

To the extent possible, the CDS software should be built to detect and recover from hardware faults. Therefore, "destructive" testing shall be included in all test procedures.

# 7 Real-time Code Generator (RCG)

The RCG is to be tested in two primary fashions:

- 1) Detection and reporting of model errors. Some test models are to be developed that have deliberate errors built in, such as missing input/output links, etc. These will be used to verify that the RCG correctly identifies and reports these errors.
- 2) Proper code generation. The CDS test and subsystem production models used in the test systems will be verified for proper operation on each code change and subsequent rebuild.

# 8 Real-time Control Software

Real-time control includes the CDS real-time core and user models compiled into single executable kernel object. This code is to be subjected to a number of primary tests, as categorized in the following subsections. In general, this code must be tested to meet the requirements defined in LIGO-T0900603.

### 8.1 Timing

Proper timing is a critical component of the real-time software. Two categories of real-time code run on each CDS real-time Front End (FE) computer, both of which require timing tests:

1) I/O Process (IOP): This process performs startup synchronization and continuous communication of data between user applications and the ADC/DAC modules. One IOP runs on each CDS FE computer.

2) User Applications: This is real-time software built by the RCG from user models. The number of user applications that may be run on a single FE computer is dependent on the number of CPU cores available. Typical loading is expected to be 4 to 6 user applications per FE computer.

#### 8.1.1 IOP

The IOP synchronizes all other real-time tasks on an FE computer. Timing tests are to include:

- 1) Proper startup synchronization. All real-time code is to start synchronous with the first ADC sample coincident with the GPS one second marker. Verification of this startup timing is to be done by making use of the duotone signal applied to one ADC channel and an IRIG-B time code receiver module and the following software:
  - a. Built in line fit routine
  - b. Matlab based duotone test software
  - c. IRIG-B card time readings
- 2) Time required to pass ADC data to user applications. The goal is <5usec, with maximum limit of 10usec. At no point shall this application run longer than 14usec, including any DAQ and other housekeeping activities.
- 3) Proper synchronization of data written to DAC modules. Requirement is that all data is written to DAC modules between 65KHz clock cycles.
- 4) Cycle to cycle code timing stability. Code execution time should not vary by more than a few useconds from cycle to cycle.
- 5) Stable timing as more user applications are added to the FE computer. IOP execution time should not change considerably as more applications are loaded on the computer.

#### 8.1.2 User Application

The user application is synchronized by the IOP. Timing tests are to include:

- 1) Proper startup synchronization. Requirement is that all real-time applications start on the ADC sample coincident with the GPS one second marker.
- 2) Long-term drift. Requirement is no drift ie application detects every ADC sample on time.
- 3) Timing of DAC writes. The user application must pass data, at the appropriate time, to the IOP for writing of data to the DAC hardware. A diagnostic exists in the IOP software for timing verification. It shall be ensured that the application DAC writes occur before the IOP time to write that information to the DAC hardware.
- 4) Cycle time stability. The time to read ADC data, run the user application and write DAC data shall not vary by more than a few useconds from cycle to cycle.
- 5) Effects of new core software. A record of timing information for standard test models will be maintained for comparison when user applications are rebuilt against new core software to detect either improvements or degradation in performance.

### 8.2 IOC

The CDS real-time core software provides interfaces to hardware mounted in IOC. A test procedure is to be developed to verify these interfaces, as listed in the following subsections.

#### 8.2.1 ADC/DAC tests

The test procedure shall verify proper operation of the IOP software in regards to proper interfacing of ADC and DAC modules:

- 1) Timing synchronization, as described previously.
- 2) Proper card and channel addressing. IOCs typically contain multiple ADC and DAC modules. It must be verified that the code addresses and reads/writes the correct modules, as designated in the user applications.
- 3) Proper module type identification. The code is required to support a number of standard I/O modules and properly map them to user applications.
- 4) Load testing ie verification that the IOP software maintains a minimum time to read/write data. The IOP is required to handle a minimum of 10 ADC/DAC modules in a single IOC.

#### 8.2.2 Binary I/O tests

Binary I/O modules are detected by the IOP, with address information passed to user applications, which subsequently read/write data to these modules. Tests are to include:

- 1) Proper detection and address passing by the IOP.
- 2) Proper read/write by user applications.
- 3) Read/write performance testing. Binary I/O modules typically take longer (as musch as a couple of useconds) per read/write operation. Therefore, this task has been delegated to the user application to keep IOP performance up.

### 8.3 Code Functional Tests

All of the code modules produced by the RCG are to be tested for proper operation on the real-time systems. Particular attention is to be given to test of the CDS standard IIR and FIR filters, which are the basic building block of control applications.

#### 8.4 Real-time Inter-Process Communications (IPC)

The CDS core software contains a standard module to perform all IPC communications, either between two processes on the same computer or processes on different computers via real-time networks. Testing is to include:

- 1) IPC between tasks between processes on the same computer.
- 2) IPC via Long Range Reflected Memory (RFM-LR). This communication is via GEFanuc PCIe reflected memory cards used to communicate real-time data between the LIGO corner station and end station computers.
- 3) IPC via Short Range Reflected Memory (RFM-SR): This communication is via Dolphinics PCIe networks between computers co-located in the MSR.

Requirements to be verified:

- 1) Proper data passing ie data is received correctly.
- 2) Time synchronization: Built-in IPC timing diagnostics are to be verified to function properly, then used to support further testing.
- 3) Load: The exact number and rates of IPCs to be supported in aLIGO are TBD. Therefore, testing will be done to determine upper performance limits.

#### 8.5 Real-time Data Acquisition

Part of the real-time code 'wrapper' is a standard data acquisition function, which runs at the same rate as the user application. Testing is to include:

- 1) Verification of proper decimation filtering. Requirement is that software properly decimates data in 2<sup>n</sup> steps from the application native rate down to a lower limit of 256Hz.
- 2) Data transmission to shared memory. Each applications must be capable of transmitting data to shared memory at up to 2MByte/sec. This is a combination of permanent DAQ channels and test point channels.
- 3) Code must properly detect GDS test point signals, from AWGTPMAN, and inject/transmit data, as appropriate.
- 4) Limits: Tests shall include DAQ and test point channel count limitations, including proper diagnostics to prevent overloading faults.
- 5) Timing benchmarks: For subsequent comparisons among code revisions, standard benchmarks will be developed and result recorded.

## 8.6 DAQ Data Transmission

A separate DAQ network transmission task is provided for each real-time application to communicate DAQ data to the DAQ system Data Concentration (DAQDC). Testing of this module is to include:

- 1) Proper timing ie data must arrive at DAQDC within a 60msec window.
- 2) Load testing: Must handle up to 2MByte/sec of data transmissions at 16Hz intervals.
- 3) Detection of and recovery from network errors, including hardware faults.

### 8.7 Arbitrary Waveform Generator / Test Point Manager (awgtpman)

A separate awgtpman task exists for each real-time application for purposes of setting test points and injecting test signals. Tests are to include:

- 1) Proper setting of test points, along with proper detection of these settings by the realtime applications.
- 2) Proper injection of test signals including signal verification.
- 3) Synchronization with real-time application. Awgtpman must be properly synchronized in time to interface properly with the real-time application.

To accommodate this testing, standard DTT test configuration files are to be developed for re-use in testing.

# 9 DAQ System

The DAQ system consists of the following major components:

- 1) DAQ Data Concentrator (DAQDC)
- 2) DAQ Frame Writer (DAQFW)
- 3) DAQ Network Data Server (NDS)

The Production Test System will be used to test DAQ hardware and software. Real-time applications, running on the various production test computers, will provide the data load.

## 9.1 DAQDC Testing

#### 9.1.1 Load Testing

The DAQDC must be capable of receiving, and subsequently broadcasting, a minimum of:

- 1) 15MByte/sec of data destined for data archival.
- 2) Additional 15MByte/sec of test point data ie temporary channel data.
- 3) Up to 100K slow (EPICS) channel data.

### 9.1.2 Timing

The DAQDC must run synchronous with the real-time DAQ applications, though timing requirements are much less stringent (measured in msec instead of µsec). Testing is to include:

- 1) The DAQDC must receive, combine and broadcast DAQ data within a 62msec window (16Hz).
- 2) DAQDC must detect and report late data transmissions by real-time computers.
- 3) Time information is to be provided by an IRIG-B receiver card installed in the DAQDC computer. As part of the testing, the interface software to this module shall be tested along with proper time verification.

#### 9.1.3 Data Error Detection

All real-time systems include a CRC checksum as part of their data transmission. The DAQDC must run its own CRC check on the data and compare to the sent CRC. A test shall be developed to verify that both:

- 1) DAQDC properly detects the error.
- 2) DAQDC properly reports the error.

### 9.1.4 EPICS Channels

The DAQDC is responsible for acquiring EPICS channel data from the FE control Ethernet. A test procedure is to include:

1) Load limits. Goal is provide ability to record all aLIGO EPICS channels (estimated at about 100K channels) at a one Hz rate.

### 9.1.5 Data broadcast

The DAQDC combines all of the DAQ data and broadcasts it, every sixteenth of a second, to all subsequent DAQ computers. Testing shall include verification that data broadcasts perform properly at rates >=30Mbytes/sec.

# 9.2 Data Storage

DAQFW are responsible for:

#### LIG0

- 1) Receiving DAQ data broadcasts from the DAQDC.
- 2) Organizing the data into standard LIGO Frame format, using the LDAS supplied Frame.cpp library.
- 3) Performing data compression, employing the compression algorithm provided as part of the LIGO Frame library.
- 4) Writing the Frame data to disk.

Testing to be performed on the DAQFW software includes:

- 1) Proper data reception and error detection.
- Proper data formatting, compression and writing at >15Mbytes/sec to 32 second Frame files.
- 3) Data compression and storage time. At present, the Frame file covers 32 seconds of data. Therefore, all data compression and writing must be completed within that time frame. As part of this testing, longer frame file times will be tested for comparison.

NOTE: Because of the anticipated large number of EPICS channels and the high, non-compressible Frame overhead for each channel, it is still TBD if EPICS channels will or will not be written to Frames independent of the fast data channels with file lengths on the order of minutes.

#### 9.3 Data Distribution

CDS requirements include the ability to send 'real-time' or archived data, on request, to support software running on CDS control room computers. This functionality is provided by the Network Data Server (NDS) software, which runs on separate computers in the DAQ system.

Testing is to include:

- 1) Proper network connections (interface) to all supported CDS control room tools, such as dataviewer, DTT, Matlab, etc.
- 2) Data validation ie NDS accurately transmits both real-time and archived data.

### **10 Integrated System Tests**

Integrated system testing is to be performed on the production test system. This system contains a fair fraction of the total computing and networking equipment associated with a single IFO and therefore lends itself to detecting software/hardware problems associated with large scale systems. For example, effects on various code due to high network usage, high I/O rates, high channel counts and rates of IPC, etc.

Using this system, various integrated operational tests are to be performed:

- Interaction of many real-time applications. Each of the nine FE computers in the system can be loaded with as many as nine real-time applications at once to test for adverse interactions under heavy loading.
- Large scale IPC transmissions. The system contains both RFM-LR and RFM-SR to accommodate these tests.
- Heavy DAQ traffic. With the number of FE computers available, it is possible to load the DAQ system at well beyond the 15MByte/sec DAQ plus 15MByte/sec GDS data traffic.

Operator computer systems may also be added to run and verify the system using the various operator interface software components. Operator interface software, such as DTT and dataviewer, along with scripts, will be employed to both verify data from the system and heavily load the CDS system with data requests to find break points.

# **11 Test Procedures**

The following table describes the individual test procedures to be developed.

Test Procedure	Description					
ADC/DAC Test	Characterize all ADC/DAC modules to be installed in aLIGO					
Real-time Software TIming	Verify timing and timing diagnostics of real-time software					
Real-time IPC	Verify performance and timing of real-time Inter-Process Communications (IPC) via all IPC interfaces.					
Real-time DAQ	Verify real-time Data Acquisition (DAQ) from ADC input to data Frames					
EPICS DAQ	Verify DAQ properly acquires EPICS data.					
DAQ Core	Test DAQ Data concentrator and Framewriter software.					
Real-time GDS	Verify proper operation of real-time code with AWGTPMAN.					
Network Data Server	Test operation of DAQ NDS.					
GDS	Test of GDS tools supported by CDS for control room use.					
Compatibility Testing	Verify that interfaces are maintained to existing operations support tools, such as awgstream, Matlab , scripts, etc.					
CDS Diagnostics	Verify operation of all built-in CDS diagnostics.					
Fault Tolerance	Procedure(s) to test system tolerance to hardware fault conditions.					
Code Generator	Verify that RCG produces proper real-time source code and proper compilation.					
	Provide a test case for each RCG 'part', such that individual code components may be verified.					
Signal Integrity	Verify data at source matches data at archive including verification that DAQ system does not introduce any data anomalies.					