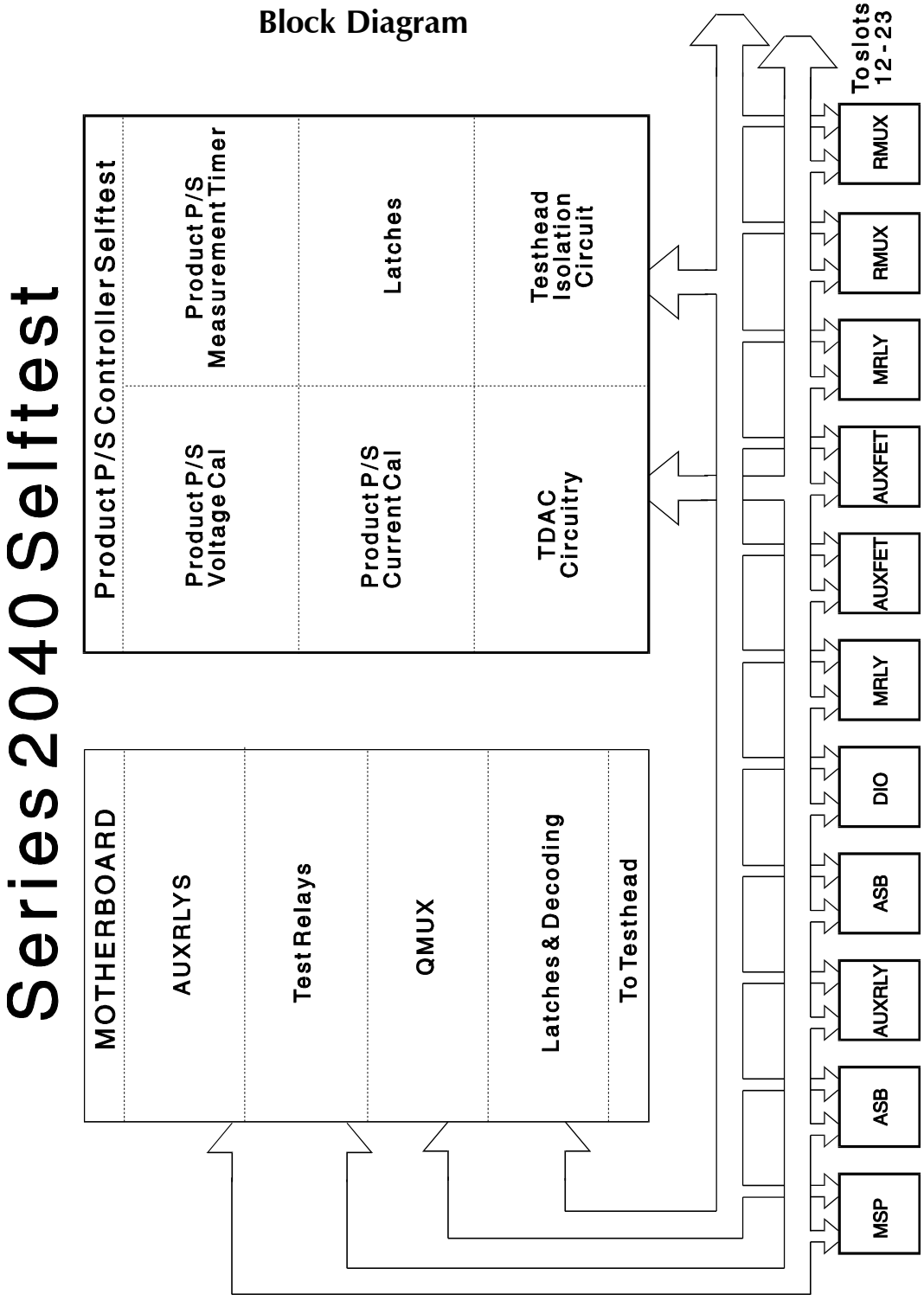
Series 2040 Test Systems

2040 Functional Selftest

Part Number #4200-0170
Version 2.1

Table of Contents

2040 Functional Selftest


Block Diagram
Series 2040 Selftest
Product P/S Controller Selftest
Product P/S Measurement Timer
Product P/S Voltage Cal
Latches
Product P/S Current Cal
Testhead Isolation Circuit
TDAC Circuitry
MOTHERBOARD
AUXRLYS
Test Relays
QMUX
Latches & Decoding
To Testhead
To slots 12 - 23
RMUX
RMUX
MRLY
AUXFET
AUXFET
MRLY
DIO
ASB
AUXRLY
ASB
MSP

Selftest Assembly

The Digalog Selftest Assembly is a Patchboard hardware interface that is installed in place of the UUT fixture. It contains the precision references, loads, and switching needed to statically and dynamically test every module in the test system. The precision references are based on a Temperature Compensated Digital to Analog Converter (TDAC). This device is verified to NIST (National Institute of Standards and Technology) standards during the certification procedure.

The functional selftest procedure is a fully automated routine of individual tests for every separate module in the test system. These individual tests include both analog and digital electronics. In addition to the individual tests geared to each individual module, a battery of test procedures is provided to verify the entire system. These tests include switching, analog and digital sources, and the measurement system.

The individual tests that comprise the functional selftest procedure are discussed in the next sections of this manual. Each individual program is briefly discussed, and a short failure analysis is provided listing probable causes. These programs include both the standard Selftest routines and the new Turbo Selftest routines. Differences in the tests for both types of Selftest Assemblies are explained when applicable. Since each 2040 Test System can be configured differently, the individual Selftest routines available for each system will vary. It is important that the boards present in the Selftest Assembly match the boards present in the Testhead. The contents of the Turbo Selftest may be verified using the Constat program by selecting the Constat icon from the Digalog program group using the Windows Explorer. For the standard Selftest Assembly, the boards must be physically checked and verified.

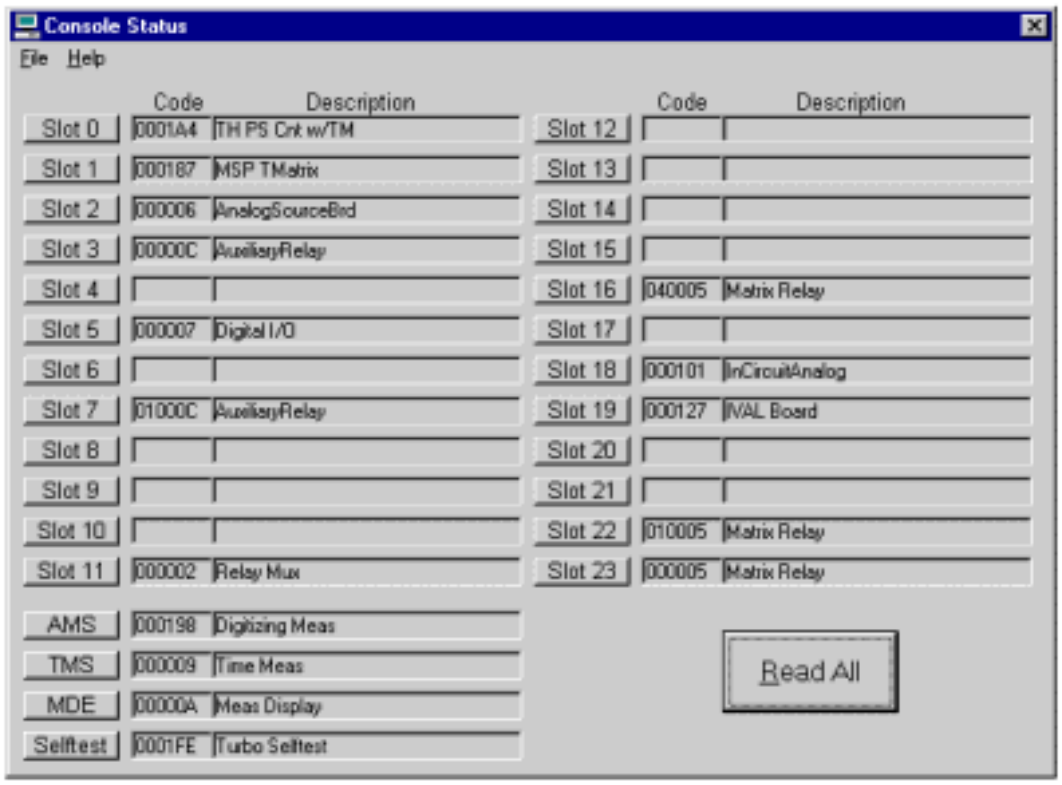
Failure to verify the boards in the Selftest Assembly could result in damage to the Selftest Assembly or the Testhead!

NOTE: If any Functional Selftest routine fails, or whenever any Testhead board is replaced, a calibration must be performed. Refer to the calibration section of the appropriate Maintenance Manual.

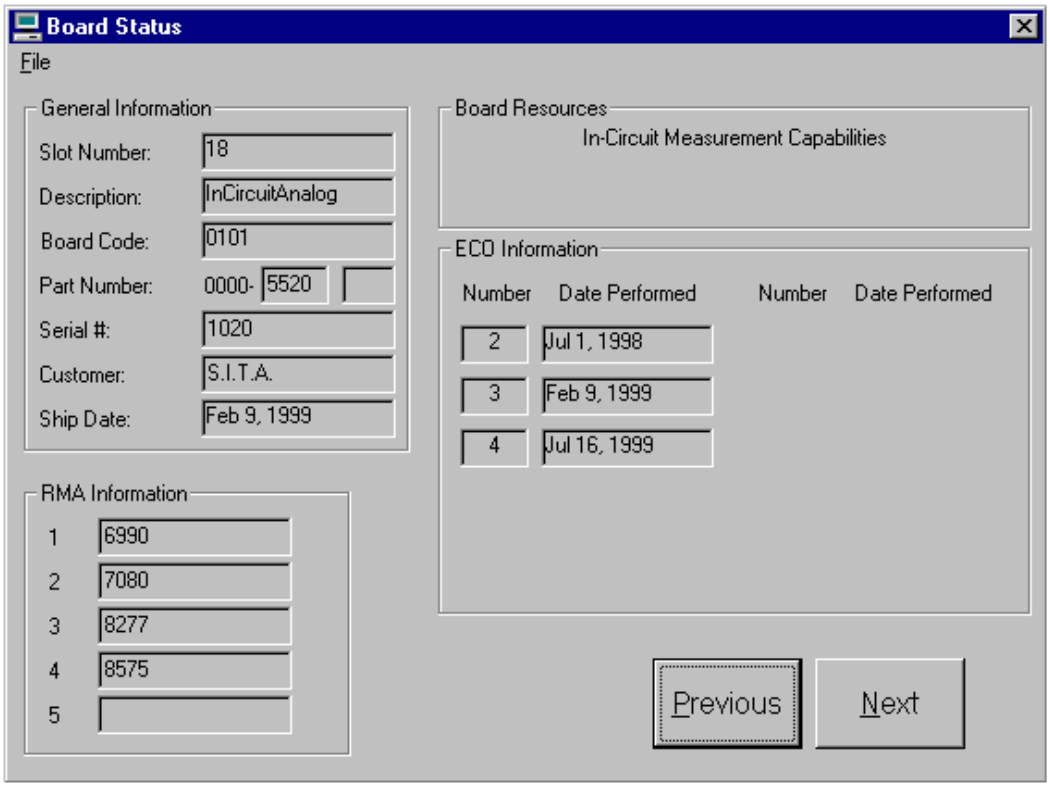
ConStat (Console Status)

Constat is a program designed to give the user a variety of information about the hardware modules that make up the Series 2040 Test System. It provides the user with the contents of the EEPROMs on the Testhead boards.

Every Testhead board in a Series 2040 Test System has an EEPROM which stores that board's history. Information such as the Digalog Part Number for the board, a board description, ECO levels and installation dates, and RMA dates is contained in the EEPROM. Constat enables the user to display or print all of the information stored in the EEPROMs. This data informs the user of the names, quantities, and locations of the boards in the Testhead (such as which slot a particular Analog Source Board occupies). It also helps in reporting troubles to Digalog Systems with a given board since its history in terms of repair and ECOs is available with only a few mouse clicks. When Constat is executed from the Digalog program group, the Console Status dialog shown below.



To select a slot, simply use the mouse to select the name of the desired slot, such as Slot 4, AMS, or Selftest. Slots are numbered from 0 to 23 with the AMS, TMS, and MDE boards in the measurement section of the Testhead. Although the Turbo Selftest Assembly is physically mounted on the Patchboard receiver, it is also read by the Constat program. After selecting a slot, the information for the selected board will be displayed using the "Board Status" dialog shown in the example below.



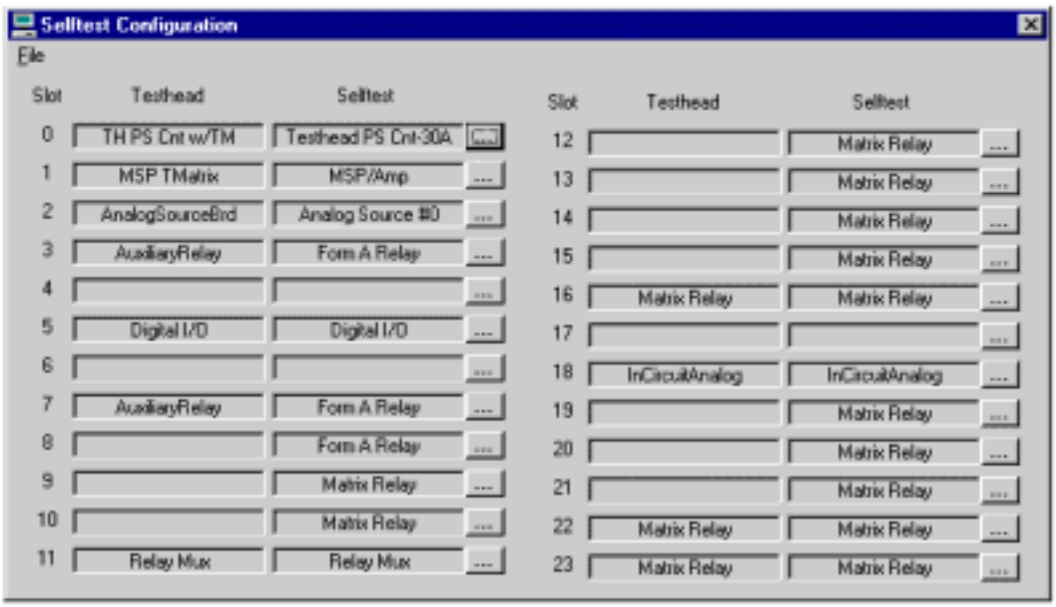
When the "**R**ead All" button is selected, the program reads every slot in the Testhead and displays the current contents of the Testhead in the "Console Status" dialog. This routine also reads the Turbo Selftest Assembly if it is installed. The Information can be printed by selecting the "Print **C**onsole" option from the **F**ile menu.

> *Note, the shipping date is contained within this information. This data is used to determine whether a board is still under warranty or not. The user can easily determine if a board is still covered for warranty repair by checking the shipping date with Constat. (Digalog warrants all boards for a period of one (1) year from the date of shipment).*

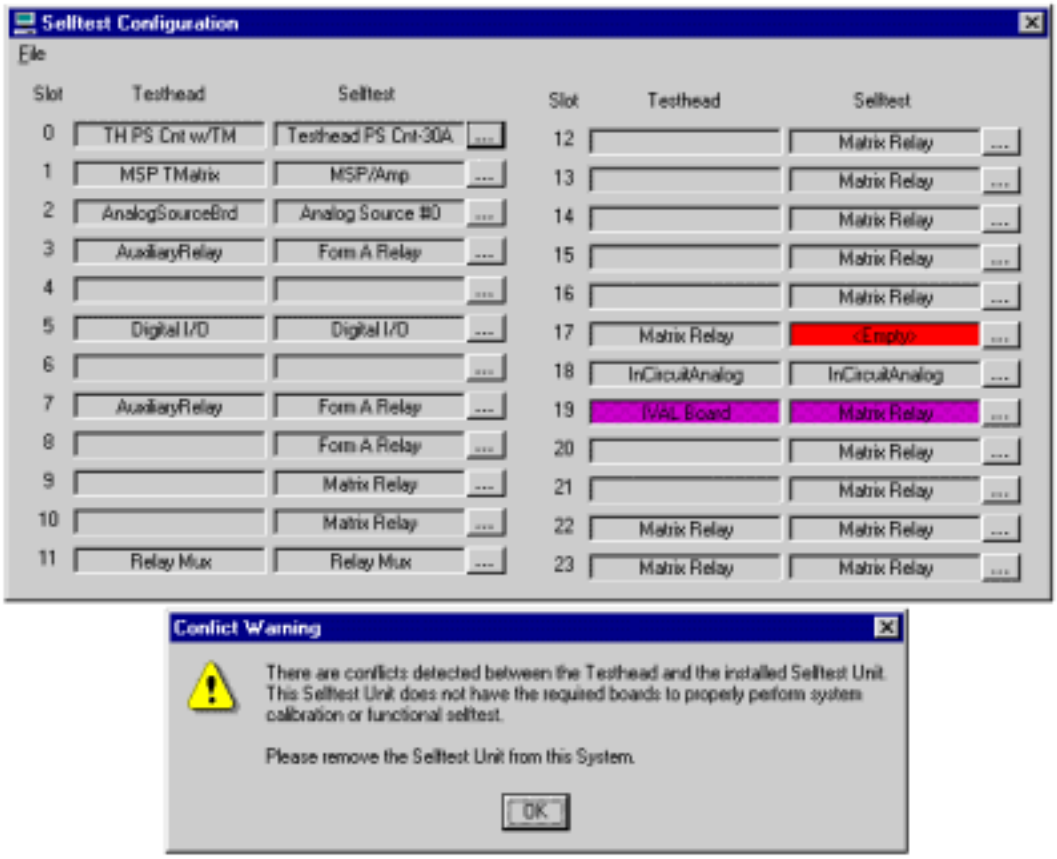
The RMA LIST is a list of Return Material Authorizations issued and processed for that board. Every time that a board is returned to Digalog for repair, this list is updated.

If the information from the "Board Status" screen needs to be printed, use the "**P**rint All Info" selection under the **F**ile menu. The "Printer **S**etup" option from the File menu is the standard Windows printer setup routine that is common to all Windows applications.

If the "Selftest" is selected from the Console Status screen, the "Board Status" information for the Selftest Assembly is displayed. If the "Config" command button is selected from this screen, the entire contents of the Turbo Selftest Assembly is displayed next to the Testhead contents for comparison. This can be used to ensure that all of the Testhead boards have a corresponding Selftest board in the Turbo Selftest Assembly.



The screen shot on the next page shows a mismatched Testhead and Turbo Selftest Assembly configuration. Note that the Matrix Relay board in slot 17 is facing an empty Selftest slot. This means that the board cannot be tested, but poses no risk of damage to the board or Selftest. An "empty"condition such as this is highlighted in red (dark grey in a black & white printout).



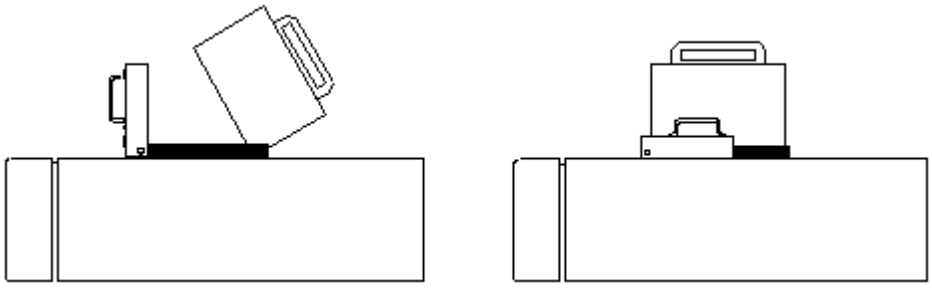
The IVAL board in slot 19 is facing a Matrix Relay selftest board. This “conflict” condition, highlighted in purple (darker grey in a black & white printout), not only means that the board cannot be tested, but also presents the possibility of damage if Selftest programs are run.

If there is any doubt about the configuration of either the Testhead or the Turbo Selftest Assembly, Constat should be run and the configurations verified before attempting to run any Selftest programs.

Selftest

Selftest on the Series 2040 Test System is an automated procedure. The programs that comprise Selftest are organized under the Selftest Executive in a test menu called “Functional.” These programs exercise the hardware functions available to the user through the functional calls. To perform Selftest, proceed as follows:

1) Install the Selftest unit on the Patchboard receiver assembly as shown below:



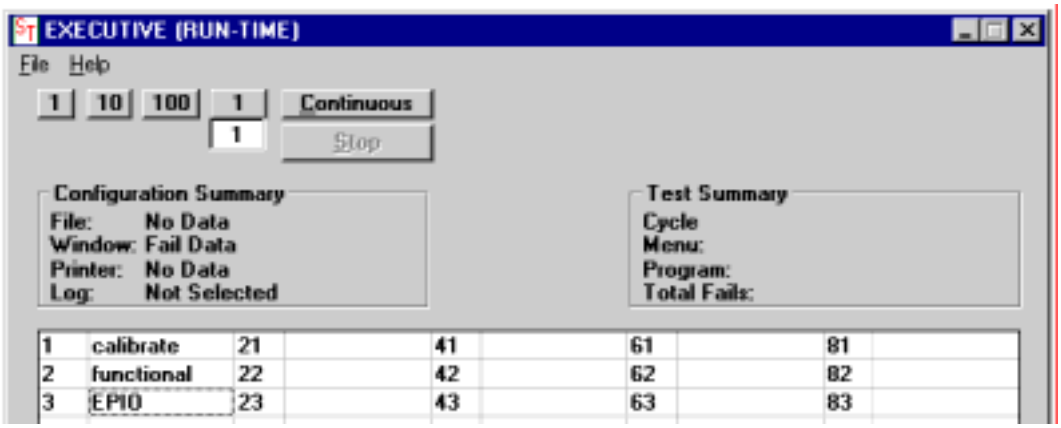
A) Pull the locking handle towards the front of the unit until it is completely vertical.

B) Slide the pins on the rear of the Selftest unit into the Patchboard receiver and rotate the Selftest unit forward until it is fully seated in the Patchboard receiver.

C) Install the Selftest data cable into the receptacle behind the Selftest unit with the colored edge of the ribbon cable on the left side (opposite the locking handle).

D) Rotate the locking handle back to a completely horizontal position to lock the Selftest unit into place.

2) Enter the Selftest Executive by selecting (double-clicking on) the "SelfExec" program from the Digalog program group. The Selftest Executive menu will be displayed as shown below.

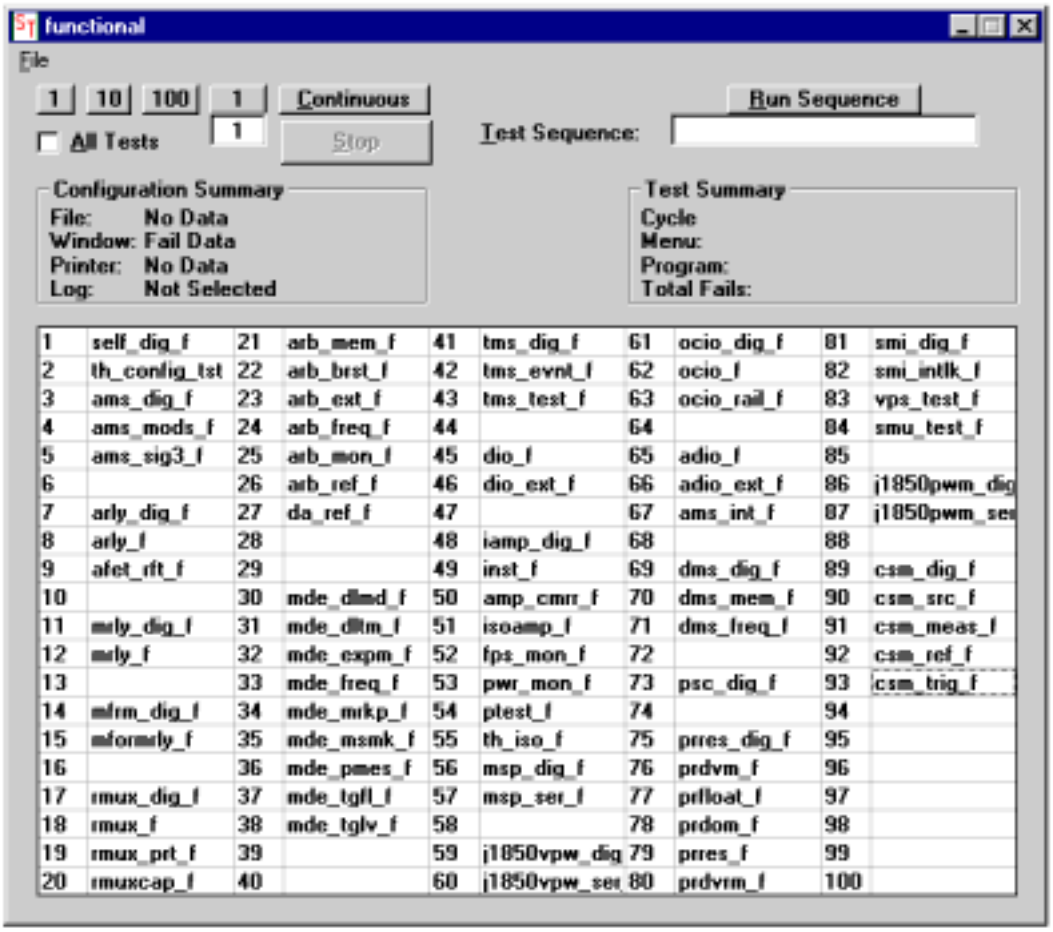


3) From the File menu, select Configure, and the "Report Utilities" dialog box will appear. This dialog controls three separate ways of reporting the data collected during the tests. Data may be stored to a file, displayed on the screen, and/or printed.

In addition, each of these three reporting functions can report "All Data", "Fail Data" or "No Data". "All Data" includes all of the pass or fail sequences from the test. "Fail Data" includes only information on the tests that failed. "No Data" disables that reporting function. To select, merely use the mouse to check or uncheck each reporting function. If a reporting device is unchecked, the options under that device appear "ghosted", indicating that they are disabled. For test purposes, enable the "Printer" and "File" options, and select the "Fail Data" option. Under the "File" section, click on the "Diagnostic File" check box. Disable the Window reporting functions by selecting the "No Data" option box.

4) Once the type(s) of reporting is completed, click on "OK" to return to the Selftest Executive menu. Select "Functional" from the Selftest Executive screen, and the Functional menu will appear as shown on the next page.

The Functional test menu has been organized so that tests closely related to each other are blocked together in groups. If only one test is to be run, use the mouse to select that particular test. If a related group of tests is to be run, they may be selected using only the mouse. For example, if the Measurement Display Electronics group is to be selected, place the mouse pointer on test #30 and hold the left button down, and drag the mouse down to test #38. The entire MDE group is now selected. If a random series of tests is desired, place the mouse pointer on the box labeled "Test Sequence". When the Windows text tool appears, click the left mouse button. At the blinking cursor, any



sequence of tests can be entered. The test numbers, however, must follow this syntax:

'**E**' (Execute) signifies the beginning of a sequence or acts as a separator between sequences.

"**:**" (Colon) signifies that the following number indicates the amount of iterations to perform of the previous sequence.
'**,**' (Comma) shares the definition of 'E' as a separator between sequences.

'**-**' (Dash) signifies inclusive or 'through'.

EXAMPLE: E1-3:5,6-100:2,1-5

This sequence executes tests 1 through 3 five times, tests 6 through 100 two times, and tests 1 through 5 once. Click on the “Run Sequence” button to execute the sequence entered in the textbox.

If a single test or a group of selected tests is to be run once, 10 times, or 100 times, use the mouse to select the appropriate button on the upper left. If the test or group is to be run at some other number, place the mouse pointer in the box to the direct left of the stop button, and the Windows text tool will appear. Click the left mouse button while text tool is displayed in the box. A blinking cursor appears, and the user can enter any number from 1 to 999 for the number of times a test should run.

ADIO

ADIO_f

ADIO_f is a selftest program designed to test the functionality of the Adjustable Level Digital I/O boards installed in the Testhead. This routine requires a Turbo Selftest Assembly.

Initially, the program checks for the presence of a Turbo Selftest Assembly. The program then searches the Testhead for ADIO boards. For each ADIO board found, the program verifies that there is a corresponding ADIO Selftest board. The test is then run as follows:

Check all bytes on the current ADIO board, with both driver clock and receiver clock in Mode 0. The Receiver Strobe will have a delay of zero. All data should get through.

Check all bytes under all masters with the RStrobe delay varied from its acceptable minimum to its maximum. All data should get through.

Checks enabling of one driver bit at a time for all bytes on each board. Each board is used as its own master. A bit is marched through the tri-state parameter in the 'addata()' functional call to enable only that bit. The data 0xFF is written and then read to verify that only one bit is at a logic '1' and that no data is in 'No Mans Land'.

Checks if the 'No Mans Land' registers can be set and cleared. The driver and receiver voltages are programmed so any data read will be in 'No Mans Land'. The drivers are then programmed to output a logic '1' level and then a logic '0' level. Each byte is then read and verified that the data is in 'No Mans Land'.

Checks the ability to vary the driver and receiver voltages and still read valid data. The logic '1' level is programmed from 15 volts to -13 volts in two volt increments. The logic '0' level is always programmed to be two volts below the logic '1' level. The receiver threshold levels are always programmed to be 0.1 volts below the logic '1' level and 0.1 volts above the logic '0' level. The data 0x55 is written and checked on each byte followed by the data 0xAA. Each board is used as its own master.

- **If a failure occurs, the problem is most likely in the ADIO board itself.**
- **Since the Selftest Assembly is used during the tests, it could also be faulty.**
- **Power to the Testhead and connections should always be checked.**

ADIO_ext_f

This selftest program was written specifically to test the external clocking and strobing modes of the Adjustable Level Digital I/O board. This test requires a Turbo Selftest Assembly.

Initially the program checks for the presence of a Turbo Selftest Assembly. The program then searches the Testhead for Adjustable Level Digital I/O boards and verifies that each board has a corresponding ADIO Selftest board in the Turbo Selftest Assembly.

The test is then run as follows:

Set driver registers to latch on an external rising edge. Write to the registers, read back the data and compare.

Set the mode to external but no external driver clock is given. Returned data should be the data read back from the last test because the external driver clock is not given. Check all bytes with the receiver strobe at normal delay.

Set receivers to strobe on an external rising edge with nominal delay. Set up output data and strobe receivers with an external clock. Check receivers and compare data.

Set to the external mode but do not give the external receiver strobe. Returned data should be the data read back from the last test because the external receiver clock is not given. Check all bytes with the receiver strobe at nominal delay.

The above tests also verify that the read data is not in "No Mans Land" (NML). The logic "1" and logic "0" driver levels are 5 volts and 0 volts respectively. The minimum "1" and maximum "0" levels are 3.5 volts and 2.5 volts respectively.

- **If a failure occurs, the problem is most likely in the ADIO board itself.**
- **Since the Selftest Assembly is used during the tests, it could also be faulty.**
- **Power to the Testhead and connections should always be checked.**

AMS

AMS_dig_f

This is a selftest program written to test the digital circuitry on the Amplitude Measurement System board.

The program initially searches the system for an Amplitude Measurement System board. When one is found, the program writes to the four group multiplexers on the Amplitude Measurement System board and reads back the registers. If successful, the program then writes to the four signal multiplexers and reads back the registers. If this is successful, the program exercises the mode multiplexer and reads back the registers. If these tests are successful, the test passes.

- **If the tests fail, the problem is almost certainly on the Amplitude Measurement System board.**
- **Problems in the CPU or Testhead Power Supply Controller could also cause test failures.**
- **Power to the Testhead and connections should always be checked.**

AMS_int_f

The AMS_int_f program tests the interrupt mode of the Amplitude Measurement System board in the Testhead. The three interrupt modes tested are:

Mode 0 non-triggered burst of readings

Mode 1 triggered burst of readings

Mode 2 all readings triggered by Mark

When the test is run, all three modes are checked with readings taken in the positive and negative half cycles of the signals generated by ARB0. The results are stored in an array.

- **If failures occur, the Amplitude Measurement System board is usually at fault.**

- **Since the Analog Source board, TMUX, Measurement Display Electronics, and Time Measurement System are used in the test, they should also be checked.**

- **Power to the Testhead and connections should always be checked.**

AMS_modes_f

This program exercises the Analog Measurement System board to ensure that all signals and modes are functional. It requires the Selftest Assembly with a 9.5 volt reference, an Analog Source board, and a Relay Multiplexer board. This test should be done after calibration, but the limits are kept loose and the board is merely checked for functionality.

Initially, the program searches for the first Relay Multiplexer board and determines which channel to use. The program then verifies the presence of the Selftest Assembly, and determines if it is a standard or Turbo Selftest. The program also verifies the presence of an Analog Source board in the Selftest Assembly. The multiplexers are then set to the proper voltage, and the TDAC is initialized. The test relays in the Selftest Assembly are engaged, the multiplexers are selected, and the TMUX on the Isolation Amplifier board (or the TMUX on the Multiple Serial Protocol board) is programmed. The TDAC is used as a source for Modes 0, 1, 3, and 4. The Analog Source board is used to generate a sinewave for Mode 2. The Relay Multiplexer board is used for Sig1, Sig2, and Sig4, while the TMUX is used for Sig3.

- **If the test fails, the Amplitude Measurement System board is almost always at fault.**

- **If only Mode 2 fails, the problem could be in the ARB on the Analog Source board.**

- **If only Sig3 fails, the problem could be the TMUX on the Isolation Amplifier board, the Multiple Serial Protocol board, or any other board in Slot 1 of the Testthead.**

- **Power to the Testhead and connections should always be checked.**

AMS_sig3_f

The AMS_sig3_f selftest routine checks the Sig3 path of the TMUX and Amplitude Measurement System for all five (0 - 4) modes and for voltages from -9 volts to +9 volts in 1 volt steps.

Initially, the program checks for the presence of a Selftest Assembly. Once the Selftest is found, the program turns on the Testhead power and sets up the TMUX channel on the Iboard in slot 1. The test is then run for all five modes using the TDAC as a voltage source. Note, the high and low limits for some of the five modes vary, and the program adjusts the limits according to whatever mode is being tested.

- **If the test fails, the Amplitude Measurement System board is almost certainly at fault.**
- **However, the TMUX channel on the board in slot 1 could also be faulty.**
- **Power to the Testhead and connections should always be checked.**

AuxRly Family

This section covers the AuxFET, AuxRly, High Current FET, & Power Relay boards.

AFET_rft_f

This program measures the turn-on (Ton) and turn-off (Toff) times of each FET switch on an AuxFET board. This test requires a Selftest Assembly.

Initially, this routine checks for the presence of a Selftest Assembly, and determines if the assembly is a Standard or Turbo Selftest. Then, a 10 volt source is connected to one side of each FET switch on the board through a mux channel. The other side of the switch is grounded. The original Selftest unit inserts a 300 ohm current limiting resistor between the voltage source and the FET switch, while the Turbo Selftest inserts two voltage dividing resistors, one on each side of the FET switch. The rise and fall time of each FET switch is measured as it is turned on and off respectively.

- **If a failure occurs, the problem is most likely in the AuxFET board itself.**
- **Since an Analog Source Board, Relay Multiplexer, Amplitude Measurement System, Measurement Display Electronics, and Time Measurement System are also used in the test, they should also be checked.**
- **Since a Selftest Assembly is used during the tests, it could also be faulty.**
- **Power to the Testhead and connections should always be checked.**

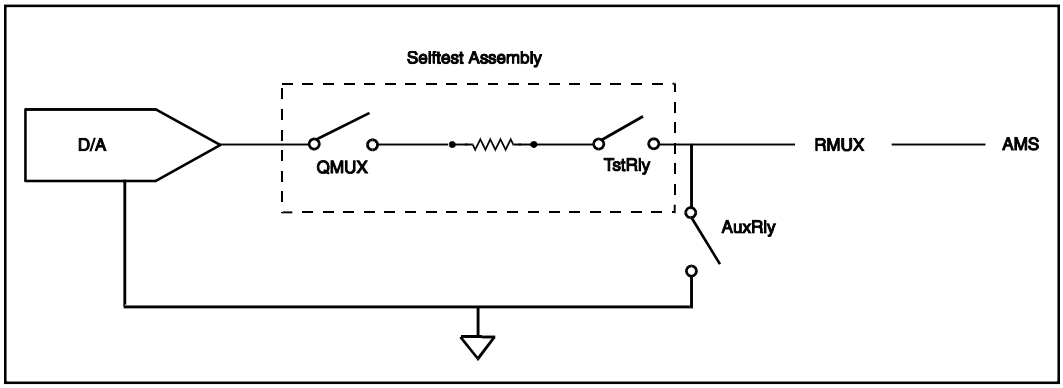
ARLY_f

ARLY_f exercises the Auxiliary Relay, Auxiliary FET, Power Relay, and High Current FET boards. It requires the Selftest Assembly and a Relay Multiplexer board. (It may also require a D/A from an Analog Source board if a standard Selftest is used). It is assumed that the measurement system is working, although it doesn't need to be calibrated.

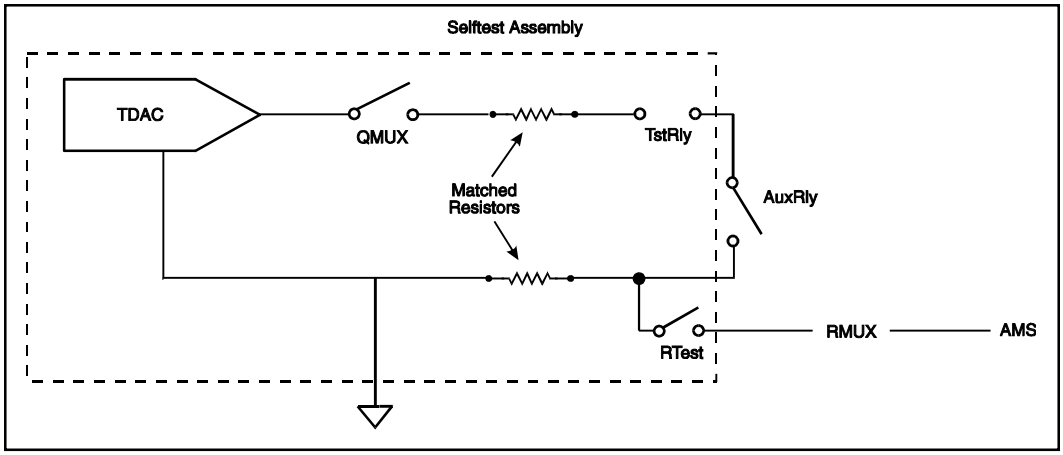
The test procedure is done in one of two ways. When a Turbo Selftest is

detected, ARLY_f calls a separate procedure (nARLY_f) to test the relay boards in an improved manner.

When operating with a standard Selftest Assembly, individual relays are connected between a reference voltage and ground. The reference voltage is generated by a D/A and routed through the Selftest Assembly. This voltage is also routed to a Relay Multiplexer channel for measurement. The relay is assumed good if the voltage read across the relay when it is closed is less than 10% of the voltage read when the relay is open.



When operating with a Turbo Selftest, the procedure differs slightly. The reference is generated by the Selftest Assembly's high-precision TDAC, making the test independent of the Analog Source board. Also, the measurement is now taken on the side of the relay opposite the reference, allowing for the complete isolation of each relay board in the Testhead. The relay is assumed good if the voltage read across the resistor when it is closed is between 45 - 55% of the reference voltage, and less than 5% of the reference is read when



the relay is open.

- **If a failure occurs while operating with a standard Selftest Assembly, the problem could be on any relay or Matrix Relay board in the system. They should be removed, one at a time, to isolate the problem to the correct board.**

- **If a failure occurs while operating with a Turbo Selftest Assembly, the problem is likely to be the relay board indicated by the error message.**

- **Since the Relay Multiplexer and Amplitude Measurement System boards and the Selftest Assembly are also used in the test, these should also be checked.**

- **Power to the Testhead and connections should always be checked.**

ARLY_dig_f

The ARLY_dig_f test merely performs multiple write and readback operations to four latches on the Auxiliary Relay, Auxiliary FET, Power Relay, or High Current FET boards in the Testhead.

If multiple boards or combinations of boards exist in the system, the program performs the test on all of them in the order indicated by the dip switch settings (0, 1, 2, etc.).

- **If the test fails, the problem is almost always on the board being tested.**

- **Power to the Testhead and connections should always be checked.**

ASB

ARB_burst_f

The ARB_burst_f program checks the burst mode of the ARB at all burst numbers. It checks the end of burst voltage to assure proper auto stop features. There are 255 tests of 2 readings each per ARB. Each test reads burst number and "end of burst voltage."

Test 1 Burst number = 1

Test 1 End of burst voltage = 0 +/-16 mv

Test 2 Burst number = 2

Test 2 End of burst voltage = 0 +/-16 mv

Note: Burst signals are all read through Sig3 using the TMUX on the Isolation Amplifier or Multiple Serial Protocol board. No Selftest is required for this test.

The program starts by searching the Testhead for the Amplitude Measurement System, Measurement Display Electronics, and Time Measurement System boards. Then the program initializes these boards. Then the program sets up the Amplitude Measurement System board and TMUX channel, and programs the first ARB on the Analog Source board to generate sinewaves. These sinewaves are routed through the TMUX to the Amplitude Measurement System board where the "end of burst voltage" is determined. The signals are then routed through the Measurement Display Electronics board where triggers are determined, and on to the event counter on the Time Measurement System board where the burst count is determined.

- **If the test fails, the Analog Source board is most likely faulty or out of calibration.**
- **Since the Amplitude Measurement System, Measurement Display Electronics, and Time Measurement System boards are also involved, they could also be faulty.**
- **If the burst count is consistently proper and the voltage fails the upper or lower limits, we may assume that the Amplitude Measurement System, TMUX, Sig3, and Time Measurement**

System are functioning properly and the problem is that the Analog Source board is not calibrated.

- **If the voltage is consistently proper and the burst count is erratic, check the Measurement Display Electronics and Time Measurement System boards.**

- **If the burst count and voltage both fail, check the Analog Source board, TMUX, and Amplitude Measurement System board.**

ARB_ext_f

ARB_ext_f is the Arbitrary Waveform Generator Start/Stop test. This program uses the event counter on the Time Measurement System board to determine if the start/stop function was completely successful.

Test 1 - Checks to see if PUL0 can generate a burst of 10 using a CPU start with external start/stop inputs high.

Test 2 - Checks to see if PUL0 can generate a burst of 10 using a CPU start with external start/stop inputs low.

Test 3 - Checks to see if CPU stop works with external start/stop inputs high on ARB0.

Test 4 - Checks to see if CPU stop works with external start/stop inputs low on ARB0.

Test 5 - Checks to see if an external start rise can generate a burst of 10 on PUL0.

Test 6 - Checks to see if an external start fall can generate a burst of 10 on PUL0

Test 7 - Checks to see if an external stop rise can stop PUL0 output.

Test 8 - Checks to see if an external stop fall can stop PUL0 output.

Test 9 - Checks to see if an external start pulse burst will start an externally disabled PUL0 output.

Test 10 - Checks to see if an external stop pulse burst will stop an externally disabled PUL0 output.]

(ARB)	ASB0	ASB1	ASB2	ASB3
test_arb	0 - 1	2 - 3	4 - 5	6 - 7
drv_arb (Pulse)	1 - 0	3 - 2	5 - 4	7 - 6
start_drv	2 - 0	6 - 4	10 - 8	14 - 12
stop_drv	3 - 1	7 - 5	11 - 9	15 - 13
pulse -out	0 - 2	4 - 6	8 - 10	12 -14

Initially, the program checks for the presence of a Selftest Assembly on the Patchboard receiver and determines if the Selftest is a standard or Turbo Selftest. Then the first Relay Multiplexer board, the Time Measurement System, and the Measurement Display Electronics are initialized and set up. The test ARB and drive ARB are also set up before the start of the first test.

All of the tests are basically run in the same manner. One ARB of a pair is used to externally trigger the other during an ARBpulse or ARBburst operation while the event counter on the Time Measurement System board is used to count the pulses and verify that the test ARB starts and stops properly. Please note, the entire Measurement System (Amplitude Measurement System, Measurement Display Electronics, & Time Measurement System) as well as the first Relay Multiplexer and all the Analog Source boards in the system are being exercised during this test. In addition, some external triggers from the ARBs are being routed through the Selftest Assembly.

- **If failures occur, the problem is most likely on the Analog Source board under test.**
- **Since the entire measurement system is being exercised, the Amplitude Measurement System, Measurement Display Electronics, and Time Measurement System boards should also be checked.**
- **The Relay Multiplexer and Selftest are being used as signal paths and could be faulty.**
- **Power to the Testhead and connections should always be checked.**

ARB_freq_f

The Arbitrary Waveform Generator Frequency test (ARB_freq_f) is designed to generate 62 different frequencies from 10 Hz to 10 MHZ using the ARBs on the Analog Source board.

The Measurement System is set up to measure these frequencies and fail any frequency that falls outside of the predetermined limits for that specific frequency. No Selftest Assembly is required for this test.

- **If the test fails, the ARB (or ARBs) on the Analog Source board is most likely faulty.**
- **However, the first Relay Multiplexer, the Amplitude Measurement System, the Measurement Display Electronics, and the Time Measurement System are also used in this test and could cause a failure.**
- **If only one ARB fails, the problem is on the Analog Source board.**
- **If a specific frequency fails on both ARBs on the same board or multiple ARBs on different boards, the problem is most likely NOT on the Analog Source boards.**
- **Power to the Testhead and connections should always be checked.**

ARB_mem_f

This selftest program tests the ARB memory on each Analog Source Board in the system.

Test 1 - Looks for addressing errors by writing a short data pattern (0x00, 0x01, 0x02, 0x03) to all addresses.

Test 2 - Checks all locations with data = address.

Test 3 - Checks all locations with data = address*.

Test 4 - Checks all locations with data = $ffff.

Test 5 - Checks all locations with data = $0000.

Test 6 - Checks the recycle bit at all locations.

Only FAIL data is output from this routine. A Selftest Assembly is not required for this test.

Initially, the program searches the Testhead for Analog Source boards. It also verifies the memory configuration for each board. Then the program performs a series of write/read operations on the memory to verify validity.

- **If errors occur, the ASB board is most certainly the problem.**
- **Power to the Testhead and connections should always be checked.**

ARB_mon_f

The ARB_mon_f program tests all D/A converters and ARBs from -15 to +15 volts in 1 volt steps. All voltages are read through the TMUX and Sig3.

Test 1 - Read the on-board +10.000 reference (ref. = 10.000 +/-.020V)

Test 2 - Read the on-board ASB ground (gnd = 0.000 +/-.020V)

Test 3 - 436 tests - 31 voltage readings each on 12 D/As and 2 ARBs

D/A reading = Programmed value +/-.025 volts

ARB reading = Programmed value +/-.035 volts

Initially the program runs a TClear and sets up the Amplitude Measurement System board. Then it searches the Testhead for Analog Source boards. The internal reference and ground are checked, and the D/A converters are programmed to various voltages. The output from the D/As is measured by the Amplitude Measurement System board through the TMUX and Sig3. When the D/As have been tested, the ARBs are programmed to specific voltages with the output being read in the same manner as the D/As. This process continues for

each Analog Source board if more than one is present in the system.

- **If failures occur, the problem is most likely in the Analog Source board.**
- **Since the TMUX and the Amplitude Measurement System board are also used in this test, they should also be checked.**
- **If multiple Analog Source boards fail, check the TMUX and Amplitude Measurement System board first.**
- **Power to the Testhead and connections should always be checked.**

ARB_ref_f

The ARB_ref_f program is the arbitrary waveform generator external reference test. It uses the other ARB on the board to drive the tested ARB. The driving ARB is set to 5 volts and applied to the reference of the tested ARB. The tested ARB voltage is varied from 0 to +15 volts in 1 volt steps, and a reading is taken.

Test 1 - 16	test ARB = 0 drive ARB = 1
	test ARB = 1 drive ARB = 0
	test ARB = 2 drive ARB = 3
	test ARB = 3 drive ARB = 2
	test ARB = 4 drive ARB = 5
	test ARB = 5 drive ARB = 4
	test ARB = 6 drive ARB = 7
	test ARB = 7 drive ARB = 6

In addition, the driving ARB is programmed to a 5 volt sinewave with 5 volts of DC offset at 500 Hz and a RMS-AC reading is taken.

Test 17	test ARB = 0 drive ARB = 1
	test ARB = 1 drive ARB = 0
	test ARB = 2 drive ARB = 3
	test ARB = 3 drive ARB = 2
	test ARB = 4 drive ARB = 5
	test ARB = 5 drive ARB = 4

test ARB = 6 drive ARB = 7
test ARB = 7 drive ARB = 6

All boards are tested with the Turbo Selftest Assembly. The standard Selftest Assembly limitations only allow for the first four ARBs.

Initially, the program searches the Testhead for Analog Source boards and sets up the Analog Source board structure so the board addresses and the TMUX offsets are known. Next, the program checks for the presence of a Selftest Assembly, and determines if the Selftest is a standard or Turbo type. The measurement system is then set up, and the tests on the ARBs begin.

- **If failures occur, the Analog Source board is usually at fault.**
- **Since the TMUX, Amplitude Measurement System, and Measurement Display Electronics boards are also used during the test, they should also be checked.**
- **Power to the Testhead and connections should always be checked.**

D/A_ref_f

The D/A_ref_f test checks the functionality of the external references of the D/A converters. It uses DA0 to DA3 on each Analog Source board as a source for the external references of the four D/A pairs that follow. The test D/As are set to +5 volts using the on-board +10 volt reference. The external reference D/As are then engaged and are incremented in steps from -10 volts to +10 volts.

SOURCE	TESTING
DA0	DA4, DA5
DA1	DA6, DA7
DA2	DA8, DA9
DA3	DA10, DA11

The output of D/A 4 through D/A11 is continually checked for appropriately shifting voltages. This test runs on a Turbo Selftest Assembly only.

- **If the test fails, the Analog Source board is usually the problem.**

- **Since the TMUX (Sig3) and the Amplitude Measurement System board are also used during the test, they should also be checked.**

- **Power to the Testhead and connections should always be checked.**

CSM

CSM_dig_f

The CSM_dig_f selftest program tests the digital logic on all of the CSM boards in the system. The program starts with a walking 1's test on the CSM's control registers. Then, each CSM channel's FPGA is tested to verify that is is programmed properly. The final part of the test exercises each CSM channel's RAM and ROM chips using selftest routines already embedded in ROM. A Selftest Assembly is not required for this test.

- **If the test fails, the CSM board is almost certainly at fault.**
- **Power to the Testhead and connections should always be checked.**

CSM_meas_f

The CSM_meas_f selftest routine tests the functionality of each of the CSM's measurement channels. The program checks each channel at 30 points across the TDAC's full scale range. A Turbo Selftest Assembly is required for the test

- **If the test fails, the problem is most likely on the CSM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

CSM_ref_f

The CSM_ref_f selftest routine tests the functionality of the Internal and External voltage references on the CSM board. The test cycles through each channel, verifying that the voltage reference is available to the output DAC as well as the HC11's own internal ADC. A Turbo Selftest Assembly is required for this test.

- **If the test fails, the problem is most likely on the CSM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**

- **Power to the Testhead and connections should always be checked.**

CSM_src_f

The CSM_src_f selftest routine tests the functionality of each of the CSM's source channels. The program checks each channel at 30 points across the TDAC's full scale range. A Turbo Selftest Assembly is required for the test

- **If the test fails, the problem is most likely on the CSM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

CSM_trig_f

The CSM_trig_f routine tests the functionality of the following components:

- Window detector inputs (measures the pulse width of a square wave at varying voltages and frequencies).
- Trigger outputs (sends out a pulse that's detected by the Turbo Selftest Assembly).
- Internal/External clock interface (The frequency is digitally divided down in the Turbo Selftest Assembly and measured by the TMS).
- Sync output (sets and clears the sync output, monitored by the Turbo Selftest Assembly).

A Turbo Selftest Assembly is obviously required for this routine.

- **If the test fails, the problem is most likely on the CSM board.**
- **A failure in the Internal/External clock interface could be caused by the TMS board.**
- **Since the Turbo Selftest Assembly is required for this test, it could**

also cause the failures.

- **Power to the Testhead and connections should always be checked.**

DAC32

DAC32_dig_f

The DAC32_dig_f selftest program tests the digital logic on all of the DAC32 boards found in the Testhead. A Turbo Selftest Assembly is required for this test.

This program will walk a logic '1' through each of the write registers and verify the data input latches by reading back the corresponding data readback register. The values of 0x00 and 0xFF will also be used in the test. This test will also verify any buried registers in the decode FPGA in a similar manner. For each test, a known pattern is written to the register. Then the inverted data is written to a different register. This is done to modify the data bus and prevent any capacitive loads from maintaining the original data. Finally the register that was written to is read and verified for the correct data.

- **If any failures occur during this test, the problem is most likely on the DAC32 board.**
- **Since the AMS board is used during this test, it could also cause a problem.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

DAC32_f

The DAC32_f selftest program tests each of the DAC32 channels on each DAC32 board found in the Testhead. A Turbo Selftest Assembly is required for this test.

The DAC32 channel output is loaded by the DAC32 selftest load resistor. First, the DAC32 selftest board Sig4 measurement paths are calibrated by stepping TDAC through its voltage range and calculating gain and offset constants using the least squares method. Next, each DAC32 channel is stepped through its voltage range and corresponding AMS Sig4 readings taken. The AMS readings are adjusted using the TDAC calibration constants, compared to the programmed values, and are displayed along with the pass/fail criteria. The

DAC32 output relay is then tested by opening the output relay and taking another AMS reading. The measured AMS voltage should be approximately zero volts. This process is repeated for each DAC32 board in the Testhead.

- **If any failures occur during this test, the problem is most likely on the DAC32 board.**
- **Since the AMS board is used during this test, it could also cause a problem.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

DAC32_ref_f

The DAC32_ref_f selftest program tests the external reference inputs on all of the DAC32 boards found in the Testhead. A Turbo Selftest Assembly is required for this test.

The DAC32 channel output is loaded by the DAC32 selftest load resistor. First, the DAC32 selftest board Sig4 measurement paths are calibrated by stepping TDAC through its voltage range and calculating gain and offset constants using the least squares method. Next, a buffered TDAC will be connected to the first eternal reference input through the DAC32 selftest board. The first DAC32 channel of the group associated with the external reference input pin will be set to 5 volts. The external reference input (TDAC) will then be stepped through its range (0 to 10 volts) and the DAC32 channel output voltage will be measured through Sig4. The adjusted Sig4 reading will be compared to the expected DAC32 output (5V * Vref/10V) and displayed along with pass/fail criterion. Then the test is run again with the DAC32 channel programmed to -5 volts. This process is repeated for each DAC32 external reference input on each DAC32 board in the Testhead.

- **If any failures occur during this test, the problem is most likely on the DAC32 board.**

- **Since the AMS board is used during this test, it could also cause a problem.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

DAC32_buf_f

The DAC32_buf_f selftest program tests the external buffer circuitry on all of the DAC32 boards found in the Testhead. A Turbo Selftest Assembly is required for this test.

The DAC32 channel output is loaded by the DAC32 selftest load resistor. First, the DAC32 selftest board Sig4 measurement paths are calibrated by stepping TDAC through its voltage range and calculating gain and offset constants using the least squares method. Next, a buffered TDAC will be connected to the first external reference input through the DAC32 selftest board. The last DAC32 channel of the group associated with the external reference input pin will be programmed to use the internal reference and set to 0 volts. The external reference input (TDAC) will then be stepped through its range (0 to 10 volts) and the DAC32 channel output voltage will be measured through Sig4. The adjusted Sig4 reading will be compared to the programmed TDAC voltage and displayed along with pass/fail criterion. This process is repeated for each DAC32 external reference input on each DAC32 board in the Testhead.

- **If any failures occur during this test, the problem is most likely on the DAC32 board.**
- **Since the AMS board is used during this test, it could also cause a problem.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

DAC32_tmldac_f

The DAC32_tmldac_f selftest program tests the Trigger Matrix LDAC input on each of the DAC32 boards found in the Testhead. A Turbo Selftest Assembly is required for this test.

First, the DAC32 selftest board Sig4 measurement paths are calibrated by stepping TDAC through its voltage range and calculating gain and offset constants using the least squares method. Each of the eight trigger matrix bus lines is connected to one of the eight trigger inputs on the DAC32 board. Each DAC32 is programmed to 0V with their output relays turned on. Each DAC32 channel is then pre-programmed to -1.0V. The output voltage of the DAC32 channel is measured by the AMS via Sig4 and verified to be 0V. All the board outputs are then strobed using the trigger matix bus, and the output voltages are measured and verified to be set to -1.0 volt. Each TM Bus is then tested separately by preprogramming all the DAC channels to a voltage level equal to the TM bus being tested. A single TM Bus line is then strobed, updating the four DAC channels connected to it. Each DAC32 output is measured and verified. Only the four DAC32 channels connected to the strobed TM Bus line should be updated to the preprogrammed voltage level. This test is repeated for each TM bus line. This process is repeated for each DAC32 board in the Testhead.

- **If any failures occur during this test, the problem is most likely on the DAC32 board.**
- **Since the AMS board is used during this test, it could also cause a problem.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

DIO

DIO_f

DIO_f is a selftest program to test the functionality of the Digital I/O boards installed in the Testhead.

For the mode 0 tests and the delay tests:
(Delay given in hundredths of microseconds)

TEST#	MASTER#	BYTES TESTED
1 - 24	1	0 - 23 respectively
25 - 48	2	0 - 23 respectively
49 - 72	3	0 - 23 respectively
73 - 96	4	0 - 23 respectively
97 - 120	5	0 - 23 respectively
121 - 144	6	0 - 23 respectively
145 - 168	1	0 - 23 respectively
169 - 192	2	0 - 23 respectively
193 - 216	3	0 - 23 respectively
217 - 240	4	0 - 23 respectively
241 - 264	5	0 - 23 respectively
265 - 288	6	0 - 23 respectively

For the 3-state pull-up and pull-down tests, the test byte# = the byte under test +1.

For the 3-state control short test, one byte at a time, 8 tests per byte.

TEST#	MASTER#	BYTES TESTED
1 - 32	1	0 - 3
33 - 64	2	4 - 7
65 - 96	3	8 - 11
97 - 128	4	12 - 15
129 - 160	5	16 - 19
161 - 192	6	20 - 23

Initially the program checks for the presence of a Selftest Assembly and determines if the Selftest is a standard or Turbo type. The program then searches the Testhead for Digital I/O boards. All drivers on all boards are now enabled, and the test is run as follows:

Check all bytes with both driver clock and receiver clock in Mode 0. Receiver strobe will have maximum delay. All data should get through.

Check all bytes under all masters with the delay in the rstrobe(2, delay,1) from 1.93 to 0.03. All data should get through.

Check all bytes under their own masters such that all outputs go into tri-state. No data should get through.

Check all bytes for correct installation of the pull-up resister network. Only data bit 4 should be low.

Check all bytes for correct installation of the pull-down resister network. Only data bit 7 should be high.

Checks for shorts between the 3-state controls in each byte for all bytes. Return data should be the 3-state outputs (read as high), with the single enabled bit as low (0xfe, 0xfd, 0xfb, 0xf7, 0xef, 0xdf, 0xbf, 0x7f).

- **If a failure occurs, the problem is most likely in the Digital I/O board itself.**
- **Since the Selftest Assembly is used during the tests, it could also be faulty.**
- **Power to the Testhead and connections should always be checked.**

DIO_ext_f

This selftest program was written specifically to test the external clocking and strobing modes of the Digital I/O board.

Initially the program checks for the presence of a Selftest Assembly, and determines if the Selftest is a standard or Turbo type. The program then searches the Testhead for Digital I/O boards and enables all drivers on all boards.

The test is run as follows:

Set driver registers to latch on an external rising edge. Write to the registers, read back the data and compare.

Set the mode to external with no external driver clock. Returned data should be the data read back from the last test. Check all bytes with the receiver strobe at normal delay.

Set receivers to strobe on an external rising edge with nominal delay. Set up output data and strobe receivers with an external clock. Check receivers and compare data.

Set external mode with no external receiver strobe test. Returned data should be the data read back from the last test. Check all bytes with the receiver strobe at nominal delay.

- **If a failure occurs, the problem is most likely in the Digital I/O board itself.**
- **Since the Selftest Assembly is used during the tests, it could also be faulty.**
- **Power to the Testhead and connections should always be checked.**

DMS/MDMS

DMS_dig_f

This is a selftest program written to test the digital circuitry on the Digitizing Measurement System board. A Selftest Assembly is not required for this test.

The program initially searches the system for an DIgitizing Measurement System board. When one is found, the program writes to the four group multiplexers on the Digitizing Measurement System board and reads back the registers. The program then writes to the signal multiplexers and reads back the registers. If this is successful, the program exercises the trigger multiplexer and reads back the registers. If these tests are successful, the program passes.

- **If the tests fail, the problem is almost certainly on the Digitizing Measurement System board.**
- **Problems in the CPU or Testhead Power Supply Controller could also cause test failures.**
- **Power to the Testhead and connections should always be checked.**

DMS_freq_f

The DMS_freq_f selftest routine can only be run on a MDMS board. It evaluates the sampling rate (per 2 channels) accuracy utilizing the internal Selftest mux that is only on the MDMS. Basically, it measures the sampling frequency/period generated by the DDSs for one test and then the End of Convert (EOC) for another. This is done at discrete intervals throughout the range of the board's sampling frequencies. Measurements of the sampling frequencies verify that the DDSs are functioning properly. Measurements of the EOC verify that the ADCs are also functioning properly. A Turbo Selftest Assembly is not required.

- **If the tests fail, the problem is almost certainly on the Digitizing Measurement System board.**
- **Since the MDE and TMS are used for sampling, they could also cause a failure.**
- **Power to the Testhead and connections should always be checked.**

DMS_mem_f

This selftest program tests the memory on the Digitizing Measurement System (DMS). A Selftest Assembly is not needed.

The test is performed in four parts. The first part does a walking 1's test on each of the memory chips. This checks the integrity of the data lines. The final three sections of the test verify the memory chips themselves. In the second part, the entire memory is filled with data that is equal to the address. Once all of the memory if filled it is verified that it is all correct. In the third and fourth parts of the test, 16 different data patterns are written to each memory location. After each data pattern is written, it is verified that the data was written correctly. All 16 tests are performed on all memory locations.

- **If the tests fail, the problem is almost certainly on the Digitizing Measurement System board.**

- **Power to the Testhead and connections should always be checked.**

EPIO

EPIO_Dig_f

This selftest routine checks the ability of the EPIO board to read and write to its own registers and checks the ability to load the address counter and its ability to count up to its maximum address. The register test is performed by doing a walking 1's test on each of the write/readable registers on the EPIO board. The values 0x00 and 0xFF are also used for a total of ten tests. For each test, a known pattern is written to the register. Then the inverted data is written to a different register. This is done to modify the data bus and prevent any capacitive loads from maintaining the original data. Finally the register that was written to is read and verified that the correct data was read.

The second test performs a walking ones test on the address counter. This is done by loading an address that has one bit high. This loaded address is then read back and verified that it is correct. This is done for all 24 bits that make up the address counter. The second part of the address counter test checks the ability of the counter to count. This is done by loading an address into the counter and then strobing the counter a set number of times. The address is then read and verified that it is at the correct location. The test is done this way for 16 blocks with each block being strobed 32K counts (for a total of 512K, the maximum depth of the EPIO memory).

A third test verifies that the TClear() functional call can clear the registers. The ability to clear the registers is needed to reset the board to a known state. For this test, each of the testable registers is filled with different data. Once they are all filled, they are read and it is verified that they contain the correct data. Then, TClear() is called to clear all of the registers. All of the registers are then verified that they are cleared.

This is the most basic of the EPIO selftest routines. If this test fails, any failures on the other selftests will be inconclusive. This is because, if the digital data bus has a problem, it is unknown what data was written to a register. On the other hand, if this test fails, some of the other selftests may pass depending on where the digital failure is located and what is being testing by the selftest routine that passed. In addition, if the address counter test fails, any selftests that use the address counter would also likely fail. The Selftest fixture is not needed for this test.

Sample Output

```
-------------------------------------------------------------------------------------------
R 0X3       0       TST: 2          ADR:Ox4     EXP:Ox00      ACT:Ox00
RST D       0       TST: 25257      ADR:Ox3     EXP:Ox01      ACT:Ox01
RST CL      0       TST: 25307      ADR:Ox3     EXP:Ox00      ACT:Ox00
CNTREG      0       TST: 18493      ADR:Ox6D    EXP:Ox04      ACT:Ox04
COUNT       0       TST: 18513      ADR:Ox6D    EXP:Ox17FFF   ACT:Ox17FFF
-------------------------------------------------------------------------------------------
```

'R 0X3' -> Testing the register at offset 0x03 on board zero
'ADR:0x4' -> While testing offset 0x3, offset 0x4 was verified that it was cleared
'RST D' -> While testing the ability for TClear to clear the registers, this line verifies that the test data was written to the register.
'RST CL' -> While testing the ability for TClear to clear the registers, this line verifies that the register was cleared.
'ADR:0x3' -> While testing the ability for TClear to clear the registers, the register at offset 0x3 is being verified.
'CNTREG 0' -> Performing a walking 1's test on the address counter on board zero
'COUNT 0' -> Performing the count up test on the address counter on board zero

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**

- **Power to the Testhead and connections should always be checked.**

EPIO_Mem_f

This selftest routine checks the ability to read and write to all of the memory locations on the EPIO board. There are two parts to this test. The first part checks the integrity of the address lines. This is done by writing incrementing data to all of the memory locations. This data is then read back, starting at address zero, and verified that it is still correct. The second part of the test checks the integrity of the data lines. This is done by performing a walking 1's test on each of the memory chips that are installed.

Many of the other selftest routines require the use of the on board memory. Therefore, this test should pass before running any of the other selftest routines (except EPIO_dig_f, EPIO_status_f, EPIO_vclk_f, and EPIO_serial_f). The Selftest fixture is not needed for this test.

It is highly recommended that this test only be executed when the Selftest Executive is configured to only output fail data. This is because the test

performs millions of operations, and if all test data is output to the screen (or file) it would take hours to execute instead of minutes.

Sample Output

```
-------------------------------------------------------------------------------
DMEM      0       TST: 3          ADR:Ox2       EXP:Ox02       ACT:Ox02
-------------------------------------------------------------------------------
```

Where 'DMEM 0' could be:

'DMEM 0'	-> Incrementing data test on the the Driver memory of board zero
'DWALK 0'	-> Walking 1's test on the Driver memory
'MMEM 0'	-> Incrementing data test on the Mask memory
'MWALK 0'	-> Walking 1's test on the Mask memory on board number zero
'RMEM 0'	-> Incrementing data test on the Result memory
'RWALK 0'	-> Walking 1's test on the Result memory on board number zero
'NRMEM 0'	-> Incrementing data test on the NML Result memory
'NRWALK0'	-> Walking 1's test on the NML Result memory on board number zero
'XNMEM 0'	-> Incrementing data test on the Expected NML memory
'XNWALK0'	-> Walking 1's test on the Expected NML memory on board number zero
'TSMEM 0'	-> Incrementing data test on the Timing Set Select memory
'TSWALK0'	-> Walking 1's test on the Timing Set Select memory on board number zero
'IMEM 0'	-> Incrementing data test on the Instruction memory
'IWALK 0'	-> Walking 1's test on the Instruction memory on board number zero
'JMEM 0'	-> Incrementing data test on the Jump memory
'JWALK 0'	-> Walking 1's test on the Jump memory on board number zero
‘ISMMEM’	-> Incrementing data test on the MSByte of the IO Formatter Segment Count memory
‘ISMWLK’	-> Walking 1’s test on the MSByte of the IO Formatter Segment Count memory
‘ISLMEM’	-> Incrementing data test on the LSByte of the IO Formatter Segment Count memory
‘ISLWLK’	-> Walking 1’s test on the LSByte of the IO Formatter Segment Count memory
‘DTSMEM’	-> Incrementing data test on the Driver Timing Set memory
‘DTSWLK’	-> Walking 1’s test on the Driver Timing Set memory
‘RTSMEM’	-> Incrementing data test on the Receiver Timing Set memory
‘RTSWLK’	-> Walking 1’s test on the Receiver Timing Set memory
‘ECSMEM’	-> Incrementing data test on the Execution Controller Segment Count memory
‘ECSWLK’	-> Walking 1’s test on the Execution Controller Segment Count memory
'TMMEM 0'	-> Incrementing data test on the Trigger Matrix Timing Set memory

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**

- **Power to the Testhead and connections should always be checked.**

EPIO_Status_f

This selftest routine is used to test the basic functionality of the START/STOP/CLOCK circuitry. It tests the ability of the functional calls to clear the STATUS bits coming from the EPIO Status register. From that point the routine determines whether or not the different STATUS bits (PRESTART, START, TGSTART, PRESTOP, STOP, TGSTOP, TEN, TESTEN, TGENABLE, and TIMEN) can be set correctly based on the circuit setup. Most of the bits are checked for both the conditions where they should be set and where they should not be set. A case where they should not be asserted would be if the software gives the strobe to assert the signal but a conditional signal is not yet asserted.

Since many of the other selftest routines (EPIO_f, EPIO_failadd_f, EPIO_format_f, EPIO_hsped_f, EPIO_instruc_f, EPIO_rmask_f, EPIO_timeset_f, EPIO_nml_f, EPIO_Inhibit_f, EPIO_tm_f, and EPIO_extsig_f) require proper operation of the START/STOP/CLOCK circuitry, this test must pass before executing those tests. The Selftest fixture is not needed for this test.

Sample Output

```
-------------------------------------------------------------------------------------
SETADD    0    TST: 1    ADR:Ox0    EXP:Ox00    ACT:Ox00
INCADD    0    TST: 2    ADR:Ox14   EXP:Ox14    ACT:Ox14
PSTRLO    0    TST: 3    ADR:Ox6A   EXP:Ox00    ACT:Ox00
PSTRH     0    TST: 4    ADR:Ox6A   EXP:Ox1000  ACT:Ox1000
STRLO     0    TST: 5    ADR:Ox6A   EXP:Ox1000  ACT:Ox1000
STRHI     0    TST: 6    ADR:Ox6A   EXP:Ox10DD  ACT:Ox10DD
TGSTLO    0    TST: 7    ADR:Ox6A   EXP:Ox00    ACT:Ox00
TGSTHI    0    TST: 8    ADR:Ox6A   EXP:OxD0    ACT:OxD0
PSTPLO    0    TST: 9    ADR:Ox6A   EXP:Ox00    ACT:Ox00
PSTPHI    0    TST: 10   ADR:Ox6A   EXP:Ox2000  ACT:Ox2000
STOPLO    0    TST: 11   ADR:Ox6A   EXP:Ox2000  ACT:Ox2000
STOPHI    0    TST: 12   ADR:Ox6A   EXP:Ox2022  ACT:Ox2022
TENLO1    0    TST: 13   ADR:Ox6A   EXP:Ox1000  ACT:Ox1000
TENHI     0    TST: 14   ADR:Ox6A   EXP:Ox10DD  ACT:Ox10DD
TENLO2    0    TST: 15   ADR:Ox6A   EXP:Ox3033  ACT:Ox3033
TGENLO    0    TST: 16   ADR:Ox6A   EXP:Ox1000  ACT:Ox1000
TGENHI    0    TST: 17   ADR:Ox6A   EXP:Ox10DD  ACT:Ox10DD
-------------------------------------------------------------------------------------
'SETADD'      -> Tests ability to set the address counter
'INCADD'      -> Tests ability to increment the address counter
'PSTRLO'      -> Tests if the PRESTART status bit can be cleared
'PSTRHI'      -> Tests if the PRESTART status bit can be set
```

'STRLO' -> Tests if the START status bit can be cleared
'STRHI' -> Tests if the START status bit can be set
'TGSTLO' -> Tests if the TGSTART status bit can be cleared
'TGSTHI' -> Tests if the TGSTART status bit can be set
'PSTPLO' -> Tests if the PRESTOP status bit can be cleared
'PSTPHI' -> Tests if the PRESTOP status bit can be set
'STOPLO' -> Tests if the STOP status bit can be cleared
'STOPHI' -> Tests if the STOP status bit can be set
'TENLO1' -> Tests if the TESTENABLE status bit can be cleared
'TENHI' -> Tests if the TESTENABLE status bit can be set
'TENLO2' -> Tests if the TESTENABLE status bit can be reset after a halt
'TGENLO' -> Tests if the TGENABLE status bit can be cleared
'TGENHI' -> Tests if the TGENABLE status bit can be set

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**

- **Power to the Testhead and connections should always be checked.**

EPIO_Serial_f

This selftest routine is used to check the functionality of the EPIO's serial interface. The EPIO selftest card includes a serial interface identical to the circuitry on an LFA card. The selftest card's serial interface acts as a client to the EPIO's serial interface. This test will also use the selftest card's serial interface to the Selftest assembly's motherboard.

In the first test, data is written to the selftest card through the Selftest assembly's serial interface on its motherboard. This data is latched on the selftest card and then written to the EPIO using the serial interface through the Patchboard. The data read by the EPIO is then verified that it is correct. This test is done with the data equal to 0x00, 0xFF, as well as 0x01 through 0x80 with the lone high bit being shifted up for each test (for a total of ten tests).

In the second test, data is written to the selftest card through the Patchboard using the EPIO's serial interface. This data is latched on the selftest card and then passed to the PC through the Selftest assembly's serial interface on its motherboard. The data transferred to the PC is checked against the data that was written to the selftest card by the EPIO. This test is done with the data equal to 0x00, 0xFF, as well as 0x01 through 0x80 with the lone high bit being shifted up for each test (for a total of ten tests).

This selftest should pass before running any of the other selftest routines that require the Selftest fixture (EPIO_f, EPIO_failadd_f, EPIO_format_f, EPIO_nml_f, EPIO_Inhibit_f, EPIO_hsped_f, EPIO_instruc_f, EPIO_rmask_f, EPIO_timeset_f, EPIO_tm_f, and EPIO_extsig_f). This is because any selftest routine that requires the Selftest fixture will need to serially write data to the selftest card in order to set it up for the that selftest.

Sample Output

```
-------------------------------------------------------------------------------------------
EP->ST      0       TST: 7      ADR:Ox21F9800    EXP:Ox10    ACT:Ox10
ST->EP      0       TST: 13     ADR:Ox21F9800    EXP:Ox01    ACT:Ox01
-------------------------------------------------------------------------------------------
'EP->ST 0'      -> Testing the EPIO to selftest serial communication on board 0
'ST->EP 0'      -> Testing the selftest to EPIO serial communication on board 0
```

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_f

This selftest routine is used to check the ability of the EPIO to drive and receive data with the differential drivers/receivers through the Patchboard. The routine verifies that all 32 channels can drive and receive data. The EPIO's serial interface through the Patchboard must be working properly in order for this test to pass. In addition, this selftest program needs a Selftest fixture with an EPIO selftest card in the same slot as the EPIO board.

The first test checks if the EPIO's differential drivers can drive data through the Patchboard. In this test, the selftest card is setup so it will latch the data driven to it. This data is then passed back to the EPIO using the EPIO's serial interface. The data received by the EPIO's serial interface is then checked against the data driven by the EPIO's differential drivers. This test is done with the data equal to 0x0000, 0xFFFF, as well as 0x01 through 0x8000 with the lone high bit being shifted up for each test (for a total of 34 tests).

The second test checks if the EPIO's differential receivers can accept data through the Patchboard. In this test, the data is written to the selftest card through the EPIO's serial interface. This data is latched on the selftest card and then driven back to the EPIO using the differential drivers on the selftest card. The EPIO is setup to read this data into its 'result' memory. The data stored in the 'result' memory is then checked against the data originally written to the selftest card through the EPIO's serial interface. This test is done with the data equal to 0x0000, 0xFFFF, as well as 0x01 through 0x8000 with the lone high bit being shifted up for each test (for a total of 34 tests).

Sample Output

```
-------------------------------------------------------------------------------------------
EPIODR    0    TST: 2     ADR:Ox21F9800    EXP:Ox00          ACT:Ox00
EPIORV    0    TST: 68    ADR:Ox21F9800    EXP:Ox80000000    ACT:Ox80000000
-------------------------------------------------------------------------------------------
'EPIODR 0'      -> Testing the ability of the EPIO to drive data through the Patchboard on
                   board 0
'EPIORV 0'      -> Testing the ability of the EPIO to receive data through the Patchboard on
                   board 0
```

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_HSpeed_f

This selftest routine tests the ability of the EPIO's receiver circuitry to accurately fill the result memory with a stream of data supplied by the drivers and the drive memory. The data stream does not pass through the Patchboard. The on-board connection between the drivers and receivers is used. The test is performed at selected clock rates over the range of the vector clock's capabilities. The EPIO_dig_f, EPIO_mem_f, and EPIO_status_f selftest routines must pass before running this test. The EPIO selftest card is not needed for this test.

First the EPIO's drivers and receivers are both enabled. The driver memory is

then be filled up with known data. The known data will continuously cycle from zero to 255 within each of the four bytes that make up the driver memory. The vector clock will be started and the drivers will begin to output their data. Once the clock stops, the data stored in the result memory will be verified. This test is performed over a range of vector clock frequencies.

It is highly recommended that this test only be executed when the Selftest Executive is configured to only output fail data. This is because the test performs millions of operations, and if all test data is output to the screen (or file) it would take hours to execute instead of minutes.

Sample Output

```
-------------------------------------------------------------------------------------------
ADDR      0      TST: 1       ADR:Ox6D     EXP:Ox1F7       ACT:Ox1F7
100       0      TST: 11      ADR:Ox9      EXP:Ox9090909   ACT:Ox9090909
-------------------------------------------------------------------------------------------
'ADDR 0'        -> Checking the final address after executing the test at one clock rate on
                   board 0
'100 0'         -> Testing the vector clock rate of 100 Hz on board 0
```

- **If any failures occur during this Routine, the problem is most likely on the EPIO board.**

- **Power to the Testhead and connections should always be checked.**

EPIO_RecMask_f

This selftest routine tests the ability of the EPIO receiver's evaluation circuitry to accurately mask received data. The drivers and the drive memory are used to supply the data used to test the evaluation circuitry. For this test, 16 channels will drive data through the Selftest Assembly to 16 different receive channels. The test is done in two different steps. The first step uses the lower 16 channels to drive data to the upper 16 channels. In the second step, the upper 16 channels drive data to the lower 16 channels. The test is performed at selected clock rates over the range of the vector clock's capabilities. The EPIO_HSpeed_f selftest routine must pass before running this test. The EPIO selftest card is needed for this test.

First, 16 EPIO drivers and 16 receivers are enabled. The driver memory is then filled up with known data. The known data will continuously cycle from zero

to 255 within each of the four bytes that make up the driver memory. The receiver's evaluation circuitry is then setup so alternate receiver mask memory locations are filled with all zeros or all ones. This effectively alternately disables/enables the mask memory. By doing this, the routine verifies that the evaluation circuitry can run at the specified speed. The vector clock will be started and the drivers will begin to output their data. Once the clock stops, the data stored in the result memory will be verified. For this test, the data stored should only match the driven data in every other memory location. The other locations will have been masked off (store data will be 0x00000000). This test is be performed over a range of vector clock frequencies.

It is highly recommended that this test only be executed when the Selftest Executive is configured to only output fail data. This is because the test performs millions of operations, and if all test data is output to the screen (or file) it would take hours to execute instead of minutes.

Sample Output

```
-------------------------------------------------------------------------------------------
ADDR      0      TST: 1        ADR:Ox6D      EXP:Ox7FFFF      ACT:Ox7FFFF
10e6      0      TST: 11       ADR:Ox9       EXP:Ox9090909    ACT:Ox9090909
-------------------------------------------------------------------------------------------
'ADDR 0'        -> Checking the final address after executing the test at one clock rate on
                   board zero
'10e6 0'        -> Testing the vector clock rate of 10 MHz on board zero
```

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**

- **Power to the Testhead and connections should always be checked.**

EPIO_ECLK_f

This selftest program is used to verify the accuracy of the DDS by measuring the Edge Clock against the certified TMS. The Edge Clock is generated from the DDS, either straight through, multiplied up, or divided down. The Edge Clock frequencies tested are selected to vary the DDS over its full range of operation. This range of DDS frequencies is 25 MHz to 50 MHz. Since the measurable frequencies are limited by the selftest and tester hardware, the range of Edge Clock frequencies tested vary from 1.57 MHz to 3.12 MHz. To obtain this range of measurable Edge Clock frequencies the 25-50 MHz DDS is divided

down by 16.

The Edge Clock's output frequency is routed to the EXTCLK Patchboard pin. From there it's routed through the Selftest Assembly to an RMux card and finally to the TMS were the frequency is measured using the **freq()** functional call.

Sample Output

```
-------------------------------------------------------------------------------------------
ECLK   0   TST: 5   HI    1.571E + 06 RDG    1.570E + 06 LO    1.569E + 06 HZ
-------------------------------------------------------------------------------------------
```

The example line is testing the Edge Clock at 1.570 MHz.

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_TM_f

This selftest routine is used to test some of the inputs and outputs of the Trigger Matrix on one EPIO board. This test consists of two separate sections. Each of the sections tests a different input or output. An output signal is defined as a signal that is generated on an EPIO and leaves through the Trigger Matrix. An input is defined as a signal that comes on to the board from the Trigger Matrix. The EPIO_hspeed_f selftest must pass before executing this test. A Selftest fixture is not needed for this test.

TEST #1: TMSIGx Output & TMSTOPTRG Input

This test checks the ability of the TMSIGx lines coming from the Instruction memory to stop a running test. There are four TMSIGx lines in the Instruction memory and they are numbered zero to three. A different TMSIGx bit is set at four different Instruction memory locations. The Trigger Matrix is setup such that only one of the TMSIGx outputs is routed to the TMSTOPTRG input signal. The STOP circuitry is setup to stop on the TMSTOPTRG signal from the Trigger Matrix. The test is started and it should stop when the specified bit set on the TMSIGx signal being tested is reached. After the test has stopped, the

current address of the address counter is read. It is verified that this address is at the location were the TMSIGx bit was set.

TEST #2: TMSTART Output & TMSTOPTRG Input

The setup for this test is not how the TMSTART output and the TMSTOPTRG input would normally be setup to execute a test. It is only done to verify the operation of the TMSTART signal. The operation of the TMSTOPTRG input is tested in the first part of this selftest program. The Trigger Matrix is setup to route the TMSTART output to the TMSTOPTRG input. The STOP circuitry is setup to stop on a signal from the Trigger Matrix. The START circuitry is setup to start on a strobe from the CPU. The test is then started. The START signal should propagate through the Trigger Matrix to the TMSTOPTRG signal and stop the test. It is then verified that the address counter is at the beginning of the test instead of being clocked to the end.

Sample Output

```
-------------------------------------------------------------------------------------------
TMSIG0    0    TST: 1    ADR:Ox32    EXP:Ox33    ACT:Ox33
ADDR      0    TST: 5    ADR:Ox0     EXP:Ox00    ACT:Ox00
STRTLO    0    TST: 6    ADR:Ox6A    EXP:Ox00    ACT:Ox00
STRTHI    0    TST: 7    ADR:Ox6A    EXP:Ox01    ACT:Ox01
SPADDR    0    TST: 8    ADR:OxFD    EXP:OxFD    ACT:OxFD
-------------------------------------------------------------------------------------------
'TMSIG0'     ->Testing Trigger Matrix signal #0, where the '0' could be 0-3.
'ADDR'       ->Verify the address counter did not start while testing the TMSTART signal.
'STRTLO'     ->Verify the START status bit is low while testing the TMSTART signal.
'STRTHI'     ->Verify the START status bit is high while testing the TMSTART signal.
'SPADDR'     ->Verify the address counter started while testing the TMSTART signal.
```

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**

- **Power to the Testhead and connections should always be checked.**

EPIO_ExtSig_f

This selftest program is used to verify the operation of the external START and STOP signals that are supplied by receiver channels 16-23. The EPIO_hspeed_f selftest must pass before executing this test. A Selftest fixture is required for this test. There are two different sections to this selftest routine. Both are explained below. The external WAIT and JUMP signals are tested during EPIO_instruc_f.

TEST #1: External START (Receiver Channels #16 - 23)

This part of the test checks the Extrenal Start signal using receiver channels 16 - 23 as the external inputs. For this test, the selftest unit drives data to the EPIO board. Initially, the data driven and the polarity of the Start signal are configured so the test will not start. After the EPIO is ARMed and started, it is then verified that the MAR did not increment and that the Start Status bit is not asserted. Then, data is written to the selftest (and thus driven back to the EPIO) that will cause the test to start for the current polarity setting. The address counter is then verified that it incremented to where the UnConditional END instruction is and that the Start Status bit is asserted. This test is performed on all eight External Input signals for both polarities and for both conditions were the test should and should not start.

TEST #2: External STOP (Receiver Channels #16 - 23)

This part of the test checks both of the External Stop signals using receiver channels 16 - 23 as the external inputs. For this test, the lower 16 channels drive to the upper 16 channels. Therefore, the EPIO selftest card is needed. The driver channel connected to receiver channel 16 - 23 is filled with the test data. The test is set up and then started. The driver data is clocked out and applied to the receiver channel being used as the external input. When the driven data matches the polarity of the Stop signal, the test will stop running. This test is performed on all eight external inputs, on both STOP signals, for both polarities, and for both conditions were it should stop and should not stop.

Sample Output

```
-------------------------------------------------------------------------------------------
ADS0-0    0    TST: 1    ADR:Ox0    EXP:Ox00        ACT:Ox00
SR0CL0    0    TST: 2    ADR:Ox6A   EXP:Ox00        ACT:Ox00
ADE0-0    0    TST: 3    ADR:Ox32   EXP:Ox00        ACT:Ox00
SRE0-0    0    TST: 4    ADR:Ox6A   EXP:Ox00        ACT:Ox00
RES0-2    0    TST: 13   ADR:Ox6A   EXP:OxFFFE0000  ACT:OxFFFE0000
-------------------------------------------------------------------------------------------
Testing the External START signal.
Where:
   'ADSe-s'     -> Means it is verifying the address before starting.
   'e'          -> External Input channel number being used to test the START signal
                   0 - External Input channel #16
                   1 - External Input channel #17
                   2 - External Input channel #18
                   3 - External Input channel #19
                   4 - External Input channel #20
```

5 - External Input channel #21
6 - External Input channel #22
7 - External Input channel #23

's' -> Test being performed:
0 - Don't allow the START signal to assert with the Polarity Low.
1 - Don't allow the START signal to assert with the Polarity High.
2 - Allow the START signal to assert with the Polarity Low.
3 - Allow the START signal to assert with the Polarity High.

'SReCLs' -> Means it is verifying the START status bit is cleared

'ADEe-s' -> Means it is verifying the test started by reading the address counter

'SREe-s' -> Means it is verifying the START status bit is set or cleared, depending on the test being performed

'SREe-s' -> Means it is verifying that the data stored in Result memory is correct for the current External Input being tested

```
-------------------------------------------------------------------------------------
SP0CL0    0      TST: 33      ADR:Ox6A      EXP:Ox00      ACT:Ox00
ADD0-0    0      TST: 34      ADR:Ox32      EXP:OxFD      ACT:OxFD
-------------------------------------------------------------------------------------
```

Testing the External STOP signal.
Where:

'SPeCLs' -> Means it is verifying the STOP status bit is cleared before starting

'e' -> External STOP channel number being tested
0 - External Input channel #16
1 - External Input channel #17
2 - External Input channel #18
3 - External Input channel #19
4 - External Input channel #20
5 - External Input channel #21
6 - External Input channel #22
7 - External Input channel #23

's' -> Test being performed:
0 - Don't allow the STOP signal to assert with the Polarity Low.
1 - Don't allow the STOP sig to assert with the Polarity High.
2 - Allow the STOP signal to assert with the Polarity Low.
3 - Allow the STOP signal to assert with the Polarity Low.

'ADDe-s' -> Means it is verifying the test stoped at the correct address

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_Instruc_f

This selftest routine is used to test the capabilities of the primary instructions. The instructions tested with this program are END, WAIT, DELAY, and JUMP. The Trigger Matrix secondary instructions are tested in the EPIO_tm_f selftest routine. The Enable Fail Checking secondary instruction is tested in the EPIO_failadd_f selftest routine. The Accumulator secondary instructions are tested as part of the EPIO_Carry_f selftest routine and the Utility Counter secondary instructions are tested as part of the DELAY primary instruction. The Selftest fixture is needed for this selftest.

The first group of tests verifies the functionality of the END instruction. The specific instructions included in this group are the Unconditional END, END on Equal, END on Unequal, and END Immediate instructions. To test the Unconditional END and END Immediate instructions, the END instruction is placed at a memory location. The test is started and when it stops, the memory location where the test stopped is verified that it is at the same location as the END instruction. The Unconditional END instruction is tested across multiple boards, if more than one board is present in the system.

The next tests are the END on Unequal and END on Equal tests. For each of these tests the driver and expected memory are filled with data that will cause the Equal flag to be set or reset based on if the Equal or Unequal END instruction is being tested. After each of these tests have stopped, it is verified that it stopped at the right address and that the Equal flag is set or cleared correctly for the test being done. In addition to testing, if the END instruction can stop a test, the edge clock's other two modes, lock and cycle, will also be tested with the END instruction. The lock and cycle modes are only valid for the END on Equal or END on Unequal instructions. The END on Equal and END on Unequal tests are performed across multiple boards if more than one board is present in the system.

The second group of tests verifies the functionality of the WAIT instruction. The first two tests in this section are WAIT on Equal and WAIT on Unequal. For each of these tests the driver and expected memory are filled with data that will cause the Equal flag to be set or reset based on if the Equal or Unequal WAIT instruction is being tested. The test is started and the address counter is read. The counter will be read repeatedly to verify that the test is waiting at the location of the WAIT on Equal or WAIT on Unequal instruction. Both of these

tests are performed across multiple boards.

The next two tests verify the External WAIT instructions. The external WAIT signals used for these instructions come from receiver channels 16 - 23. Since the test is in the middle of a WAIT instruction, data can not be clocked out of the driver memory to the receiver channel to change the state of the External WAIT instruction. Therefore, data has to be passed to the selftest card to change the logic level on the receiver channel being used for the external WAIT signal while the test is waiting for it to change logic level. After the test is started, the program repeatedly reads the address counter and verifies that it is waiting at the location where the External WAIT instruction is located. Once this is verified, the logic level of the receiver channel is changed by a serial write to the Selftest fixture. The address counter is then read to verify it is beyond the location where the External WAIT instruction is located. This test is only performed on one board at a time.

The third group of tests verifies the functionality of the DELAY on Counter instructions. The DELAY instruction is similar to WAIT. The difference is that this instruction is used to pause a test for a set time frame. The time frame is determined from the vector clock's frequency and the value loaded into the counter. For example, a delay of 5ms and a vector rate of 10MHZ would need a count value of 50000. For this test, the counter and vector rate are setup to cause a wait for a relatively long time frame (100's ms). The test is started and the address counter is read. The MAR will be read repeatedly to verify that the address counter is at the location of the WAIT instruction for approximately the same time frame as the delay period. This test is executed for a range of time delays, but only on one board at a time.

The fourth group of tests verifies the functionality of the JUMP instructions. The first test verifies the Unconditional JUMP instruction. For this test, the JUMP instruction will be placed at a certain location. An END instruction will be placed a few locations after this instruction and an END instruction will be placed at the location to be jumped to. Then, after the test stops, it verifies that the address counter is at the location jumped to and not at the location of the END instruction placed just after the JUMP instruction. This test is performed across multiple boards.

The next two tests verify the JUMP on Equal and JUMP on Unequal instructions. Each of these instructions are tested for the conditions where they

should jump and where they should not jump. In all cases, an END instruction will be placed just after the JUMP and at the location jumped to. Each time the test is run, the address counter will be read. It will then be verified that the address is at either the jumped to location (if the test setup was such that the jump should have occurred) or just after the JUMP instruction (if the jump should not have occurred). This test is only performed on one board at a time.

For the JUMP on External tests, the EPIO will be set up to receive data from the Selftest fixture on the external channels (16 - 23). Data is written to the Selftest fixture to set the external bit. The test will then be started. After the test has completed, it will be verified that the address counter is at the correct location. Again, these instructions will be checked for the locations where they should jump and should not jump. This test is only performed on one board at a time.

Primary instructions that have signals that are carried over the EPIO Ribbon Interconnection cable will be tested over multiple EPIO boards (if possible). By doing this, the integrity of the cable connections is verified. The instructions that are used to test multiple boards are the JUMP instruction and instructions that use the Equal flag.

Sample Output

```
-------------------------------------------------------------------------------------------
MUEND     0        TST: 1          ADR:Ox21F9800  EXP:Ox194     ACT:Ox194
-------------------------------------------------------------------------------------------
 Testing the UnConditional END Instruction (Master and Slaves)
      "MUEND"   - Current address on the Master board
      "SUEND"   - Current address on a Slave board

-------------------------------------------------------------------------------------------------------
MEIM      0        TST: 2          ADR:Ox22A9600  EXP:Ox81      ACT:Ox81
-------------------------------------------------------------------------------------------------------
 Testing the Immediate END Instruction (Master and Slaves)
      "MEIM" - Ending vector address on the Master board
      "SEIM" - Ending vector address on a Slave board

-------------------------------------------------------------------------------------------------------
MCEUMS    0        TST: 2          ADR:Ox21F9800  EXP:Ox194     ACT:Ox194
TGRUN     0        TST: 3          ADR:Ox21F9800  EXP:Ox00      ACT:Ox00
TGSTOP    0        TST: 4          ADR:Ox21F9800  EXP:Ox00      ACT:Ox00
-------------------------------------------------------------------------------------------
 Testing the Conditional END Instruction (Master and Slaves)
      "MCEiMm"  - Current address on the master board
```

```
      "SCEiMm"  - Current address on a slave board
            'i' -> Conditional END instruction
                  'U' - END-On-UnEqual
                  'E' - END-On-Equal
            'm' -> End mode
                  'S' - Stop-On-END
                  'L' - Lock-On-END
                  'C' - Cycle-On-END
      "TGRUN"   - Timing Generator is still running on the master
      "TGSTOP"  - Timing Gernerator is stopped on the master

-------------------------------------------------------------------------------------
MWUM1F  0       TST: 20    ADR:Ox21F9800  EXP:Ox154    ACT:Ox154
MWEM2F  0       TST: 25    ADR:Ox21F9800  EXP:Ox303    ACT:Ox303
-------------------------------------------------------------------------------------
 Testing the Conditional WAIT Instruction (Master and Slaves)
      "MWiMmf"  - Current address on the Master board
      "SWiMmf"  - Current address on the Slave board
      "MWiMmE"  - Ending address on a Master for a test that
            should have waited but then was continued to the end.
      "SWiMmE"  - Ending address on a slave for a test that should
            have waited but then was continued to the end.
            'i' -> Conditional WAIT instruction
                  'U' - WAIT-On-UnEqual
                  'E' - WAIT-On-Equal
            'm' -> Test Method
                  1 - The test should wait at the test address
                  2 - The test should NOT wait at the test address
            'f' -> If there is an 'F' in this location it means this board was used to force the
                  Conditional WAIT. If this location is blank, this board was NOT used to
                  force the condition. This is true for both master and slave boards.

-------------------------------------------------------------------------------------
EW1M1L  0       TST: 26    ADR:Ox10       EXP:Ox204    ACT:Ox204
EW1MLE  0       TST: 27    ADR:Ox10       EXP:Ox300    ACT:Ox300
-------------------------------------------------------------------------------------
 Testing the External WAIT Instruction
      "EWiMmp"  - Current address of the board being tested
      "EWiMpE"  - Ending address for tests that should have
            waited but were then continued
            'i' -> WAIT-On-External instruction number
                  1 - WAIT-On-External signal #1
                  2 - WAIT-On-External signal #2
            'm' -> Test Method
                  1 - The test should wait at the test address
                  2 - The test should NOT wait at the test address
```

'p' -> External signal polarity used for this test
L - Test was done with low polarity
H - Test was done with high polarity
"ADR" -> The value after ADR is the External Input channel being tested. It can have a value of Ox10 to Ox17 representing channels #16 - 23.

FWADD1	0	TST: 1	ADR:Ox200	EXP:Ox204	ACT:Ox204
FWNUM1	0	TST: 2	ADR:Ox200	EXP:Ox02	ACT:Ox02
FWA0T1	0	TST: 3	ADR:Ox200	EXP:Ox00	ACT:Ox00
FWA1T1	0	TST: 4	ADR:Ox200	EXP:Ox1FF	ACT:Ox1FF

Testing the location of failures during a WAIT Instruction
"FWADDm" - Address where the WAIT instruction is located
"FWNUMm" - Number of expected failures for the current test setup
"FWAiTm" - Address where the failure was logged at
'm' -> Specifies which test setup was used. Will be a number from 1 to 7. Each test setup has the failure at a different address.
'i' -> Index used to count through the number of expected failures. Will be a number from 0 to the number of failures displayed. Each test has a failure forced at the beginning of the vectors and within the pipeline of the WAIT instruction.

D1V1	0	TST: 38	ADR:Ox21F9800	EXP:Ox0A	ACT:Ox0A
D1V1E	0	TST: 39	ADR:Ox21F9800	EXP:OxFFFF	ACT:OxFFFF

Testing the DELAY On Counter Instruction
"DcVv" - Number of times polled where the DELAY instruction is located
"DcVvE" - Ending address of the DELAY instruction test
'c' -> Delay Counter being tested
1 - Delay Counter #1
2 - Delay Counter #2
'v' -> Number representing the counter value being used
1 - 250000 (250 mSec)
2 - 500000 (500 mSec)
3 - 750000 (750 mSec)

FDADD1	0	TST: 29	ADR:OxFFFF	EXP:OxFFFF	ACT:OxFFFF
FDNUM1	0	TST: 30	ADR:Ox6000	EXP:Ox02	ACT:Ox02
FDA0T1	0	TST: 31	ADR:Ox6000	EXP:Ox00	ACT:Ox00
FDA1T1	0	TST: 32	ADR:Ox6000	EXP:Ox5FFF	ACT:Ox5FFF

Testing the location of failures during a DELAY Instruction

"FDADDm" - Ending address of this test
"FDNUMm" - Number of expected failures for the current test setup
"FDAiTm" - Address where the failure was logged at
'm' -> Specifies which test setup was used. Will be a number from 1 to 7. Each test setup has the failure at a different address.
'i' -> Index used to count through the number of expected failures. Will be a number from 0 to the number of failures displayed. Each test has a failure forced at the beginning of the vectors and within the pipeline of the DELAY instruction.

MUJUMP 0 TST: 50 ADR:Ox21F9800 EXP:Ox2509 ACT:Ox2509

Testing the UnConditional JUMP Instruction (Master and Slaves)
"MUJUMP" - Master's address where the UnConditional JUMP was to
"SUJUMP" - Slave's address where the UnConditional JUMP was to

---

MCJUM1 0 TST: 51 ADR:Ox21F9800 EXP:Ox2509 ACT:Ox2509
MCJEM2 0 TST: 54 ADR:Ox21F9800 EXP:Ox63 ACT:Ox63

Testing the Conditional JUMP Instruction (Master and Slaves)
"MCJsMm" - Current address of the master board being tested
"SCJsMm" - Current address of a slave board being tested
's' -> Letter representing the conditional JUMP instruction being tested
U - JUMP-On-UnEqual
E - JUMP-On-Equal
'm' -> Test Method
1 - The test should jump to the Jump-To-Location
2 - The test should NOT jump

JC1 0 TST: 55 ADR:Ox6A EXP:Ox186A1 ACT:Ox186A1
JC1M1C 0 TST: 59 ADR:Ox21F9800 EXP:OxFFFFFF ACT:OxFFFFFF

Testing the JUMP Carry Clear On Counter Instrcution
"JCi" - Utility counter's reading used to verify the JCx instruction counted down
"JCiMmC" - Utility counters final reading of the board being tested
'i' -> Jump-On-Carry-Clear instruction
1 - JUMP-On-Carry-Clear Counter #1
2 - JUMP-On-Carry-Clear Counter #2
'm' -> Test Method

1 - The counter should count down
2 - The counter should NOT count down

EJ1M1L 0 TST: 75 ADR:Ox10 EXP:Ox100A ACT:Ox100A

Testing the External JUMP Instrcution
"EJiMmp" - Current address of the board being tested
'i' -> Jump-On-External instruction
1 - JUMP-On-External #1
2 - JUMP-On-External #2
'm' -> Test Method
1 - The test should jump to the Jump-To-Location
2 - The test should NOT jump
'p' -> External signal polarity used for this test
L - Test was done with low polarity
H - Test was done with high polarity
"ADR" -> The value after ADR is the External Input channel being tested. It can have a value of Ox10 to Ox17 representing channels #16 - 23.

FJADD1 0 TST: 61 ADR:Ox1500 EXP:Ox3000 ACT:Ox3000
FJNUM1 0 TST: 62 ADR:Ox1500 EXP:Ox02 ACT:Ox02
FJA0T1 0 TST: 63 ADR:Ox1500 EXP:Ox00 ACT:Ox00
FJA1T1 0 TST: 64 ADR:Ox1500 EXP:Ox14FF ACT:Ox14FF

Testing the location of failures during a JUMP forward operation
"FJADDm" - Ending address of this test
"FJNUMm" - Number of expected failures for the current test setup
"FJAiTm" - Address where the failure was logged at
'm' -> Specifies which test setup was used. Will be a number from 1 to 7. Each test setup has the failure at a different address.
'i' -> Index used to count through the number of expected failures. Will be a number from 0 to the number of failures displayed. Each test has a failure forced at the beginning of the vectors and within the pipeline of the JUMP instruction.

FLADD1 0 TST: 88 ADR:Ox1000 EXP:Ox3000 ACT:Ox3000
FLNUM1 0 TST: 89 ADR:Ox1000 EXP:Ox66 ACT:Ox66
FLA0T1 0 TST: 90 ADR:Ox1000 EXP:Ox00 ACT:Ox00
FLA1T1 0 TST: 91 ADR:Ox1000 EXP:OxFFF ACT:OxFFF
FLA2T1 0 TST: 92 ADR:Ox1000 EXP:OxFFF ACT:OxFFF
FLA3T1 0 TST: 93 ADR:Ox1000 EXP:OxFFF ACT:OxFFF
FLA4T1 0 TST: 94 ADR:Ox1000 EXP:OxFFF ACT:OxFFF

```
FLA5T1      0       TST: 95     ADR:Ox1000     EXP:OxFFF     ACT:OxFFF
-------------------------------------------------------------------------------------------
Testing the location of failures during a looping JUMP operation
        "FLADDm" - Ending address of this test
        "FLNUMm" - Number of expected failures for the current test setup
        "FLAiTm" - Address where the failure was logged at
                'm' -> Specifies which test setup was used. Will be a number
            from 1 to 7. Each test setup has the failure at a
            different address.
                'i' -> Index used to count through the number of expected
            failures. Will be a number from 0 to the number of
            failures displayed. Each test has a failure forced
            at the beginning of the vectors and within the loop
            created by the JUMP back instruction.
```

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_Timeset_f

This selftest routine is used to verify the resolution of a timing set. The Selftest fixture is required for this test.

For this test, the driver and receiver timing sets are setup to correctly clock data out from the driver memory and into the result memory. The data that was written into the result memory is verified. The active region of the driver's timing set is then changed by one edgeclock period for each test. For some of the tests, the correct data written to the result memory will not be the data driven out by the driver. This is because the driver and receiver timing sets are not in sync. By doing this, the ability of timing sets to correctly output and receive patterns can be verified as well as the ability to move the placement of the clock edges.

Sample Output

```
ADDR      0       TST: 1      ADR:Ox6D          EXP:Ox7FFF    ACT:Ox7FFF
TSET1     0       TST: 4      ADR:Ox20D9800     EXP:Ox2020000 ACT:Ox2020000
```

'ADDR 0' -> Verify the address counter incremented to the end on board number zero
'TSET1 0' -> Testing Timing Set configuration #1 on board number zero, where the '1' could be 0-2. Timing Set configuration numbers 0 and 1 will have the driver and receiver timing sets configured such that they will correctly clock data out from the driver memory and into the result memory. Configuration #2 will have the receiver strobe after the active region of the driver data. Therefore, the driver data will not be strobed into the result memory.

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_Carry_f

This selftest routine is used to verify the ability of the algorithmic units to carry logic across one or more boards. The Selftest fixture is not needed for this test.

During this test, the accumulators will be setup so they will linearly add data (instead of looping on a few addresses). By doing this, all of the output vectors will be stored into the result memory. Therefore, all of the vectors can be read back and verified. This test will be performed for a multiple of test setups. One group of tests will have each of the eight bit accumulators on one board carry over to each of the other three accumulators on that board. Then, the second group of tests will have each accumulator on a board carry over to all four accumulators on a second board. This is done to verify the four global carry lines leaving each accumulator and going through the EPIO Ribbon Interconnect Cable.

It is highly recommended that this test only be executed when the Selftest Executive is configured to only output fail data. This is because the test performs millions of operations, and if all test data is output to the screen (or file) it would take hours to execute instead of minutes.

Sample Output

```
ADDOUT   0    TST: 1    ADR:Ox110    EXP:Ox113    ACT:Ox113
ADDIN    0    TST: 1    ADR:Ox110    EXP:Ox113    ACT:Ox113
O2 I3    0    TST: 2    ADR:Ox0      EXP:OxF0     ACT:OxF0
```

'ADDOUT' -> Verify the ending address on the board with the Carry Out Accumulator
'ADDIN' -> Verify the ending address on the board with the Carry In Accumulator
'O2 I3' -> The Carry Out Accumulator is #2 and the Carry In Accumulator is #3.

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**

- **Power to the Testhead and connections should always be checked.**

EPIO_FailAdd_f

This selftest routine is used to verify the fail address latch and the fail counter. The Selftest fixture is required for this test.

For this test, the driver and expected memory will be loaded with data that will cause failures. During this routine, the last 16 channels are used to drive the first 16 channels, and vice versa. The test will be run and the fail counter and latched failed addresses are read. The number of failures that occurred will be verified as well as the failed addresses. This test will be run several times. One of the tests will be setup so there will be no failures and the other tests will be run with a range of failures.

It is highly recommended that this test only be executed when the Selftest Executive is configured to only output fail data. This is because the test performs millions of operations, and if all test data is output to the screen (or file) it would take hours to execute instead of minutes.

Sample Output

```
NF 0-1   0    TST: 2    ADR:Ox21F9800   EXP:Ox02    ACT:Ox02
FA 0-1   0    TST: 3    ADR:Ox21F9800   EXP:Ox32    ACT:Ox32
FD 0-1   0    TST: 4    ADR:Ox21F9800   EXP:OxCD    ACT:OxCD
```

'NF' -> Expected number of failures
'FA' -> Address were a failure occurred
'FD' -> Data at the failed address

'0-1' -> Byte #1 on board #0 is were the failure was forced to occur

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_Format_f

This selftest routine is used to verify the output formats of a driver channel. The Selftest fixture is not needed for this test.

For this test, the timing set of a driver channel is setup so the active region is half of the vector clock period. The timing set for the receiver is first setup to latch the data during the driver's active region. The data will then be read and verified that it matches the driver data. Then, the receiver's timing set is setup to latch the data after the driver's active region. The data will then be read and verified that it is correct for the driver output format. This sequence is performed for all eight driver output formats.

Sample Output

R0-0V	0	TST: 1	ADR:Ox0	EXP:Ox00	ACT:Ox00
R0-1V	0	TST: 2	ADR:Ox1	EXP:Ox00	ACT:Ox00

Where 'R0' could be:
- 'R0' - Return-To-Zero
- 'R1' - Return-To-One
- 'RC' - Return-To-Complement
- 'NR' - No-Return
- '/R0' - Inverted Return-To-Zero
- '/R1' - Inverted Return-To-One
- '/RC' - Inverted Return-To-Complement
- '/NR' - Inverted No-Return

Where '-0V' could be:
- '-0V' - Driving a valid logic zero
- '-1V' - Driving a valid logic one
- '-0I' - Driving an invalid logic zero
- '-1I' - Driving an invalid logic one

The valid state means the receiver's timing set should latch the data during the driver's active region.
The invalid state means the receiver's timing set should latch the data during the driver's inactive region.

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Power to the Testhead and connections should always be checked.**

EPIO_Inhibit_f

This selftest routine is used to test the Inhibit Fail mode. This mode is used to Inhibit a channel from causing an unequal condition and thus flagging a failure. This test checks each channel seperately.

For this test, 16 channels are configured to drive to the other 16 channels through the selftest. Each byte of driver and expected data is filled with incrementing data. Then, data in the channel driving to the test channel (the receiver channel) is modified to be unequal to the expected data on each vector. The first part of the test does not inhibit failures from ocuring. Therefore, after the test is run, there should be as many failures as there are vectors in the test. The received data is also verified that it is correct and it contains the opposite logic level than the expected on the test channel. Then for the second part of the test, the test channel is inhibited from flagging failures. The test is run again but this time there should be no failures logged. Once again the received data is verified that it contains the data opposite to the expected data on the test channel even though no failures were logged. This test is done for each channel as a receiver (or test channel).

It is highly recommended that this test only be executed when the Selftest Executive is configured to only output fail data. This is because the test performs millions of operations, and if all test data is output to the screen (or file) it would take hours to execute instead of minutes.

Sample Output

```
-------------------------------------------------------------------------------------------------
ADDR    0    TST: 1      ADR:Ox6D        EXP:OxFF    ACT:OxFF
FAILS   0    TST: 2      ADR:Ox22A9600   EXP:OxFC    ACT:OxFC
INH-0   0    TST: 3      ADR:Ox0         EXP:Ox01    ACT:Ox01
ADDR    0    TST: 255    ADR:Ox6D        EXP:OxFF    ACT:OxFF
FAILS   0    TST: 256    ADR:Ox22A9600   EXP:Ox00    ACT:Ox00
INH-1   0    TST: 257    ADR:Ox0         EXP:Ox01    ACT:Ox01
-------------------------------------------------------------------------------------------------
```

'ADDR' -> Verify the correct ending address for this test.
'FAILS' -> Verfify the correct number of fails occured for this test run.
'INH-0' -> Verify the correct data was written to the Result memory for the test that should log failures. The 'EXP' and 'ACT' data will be the expected and actual data stored to the result memory on the receive channels.
'INH-1' -> Verify the correct data was written to the Result memory for the test that should not log failures. The 'EXP' and 'ACT' data will be the expected and actual data stored to the result memory on the receive channels.

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Since data is being transfered through the Patchboard, there could be a Patchboard pin not making contact with the Selftest Assembly pin.**
- **Power to the Testhead and connections should always be checked.**

EPIO_nml_f

This selftest program is used to test the NML Detection capability of the EPIO. The NML Detection capability on the EPIO works on the principle that the LVDS receivers on the EPIO will drift high if they are not driven from an LFA or selftest card. There are two LVDS receivers per channel on an EPIO. One is the normal receiver. The other is for NML Detection. The two LVDS receivers are configured such that they will read opposite logic levels when a logic level is driven from the LFA or selfetst card. However, if an LVDS driver on the LFA or selftest card is disabled, the output from both receivers on the EPIO will drift high. The logic highs on both of these receivers on the EPIO flags a NML condition. In normal operation, the LFA disables the drivers to the EPIO when

it detects a NML condition.

Since the selftest card does not have the circuitry that can generate various voltage levels, nor does it have the 'voltage window' detection capability like an LFA would have, it can not cause a NML condition. To simulate a NML condition, the control channels (28-31) which are used to control the channel direction on channels 0-27, can be used to disable the selftest's LVDS drivers to the EPIO. Therefore, the receivers on the EPIO will not be driven and both their outputs will drift high. The EPIO will then log this vector as being in NML.

For this test, channels 0-27 are configured as receivers and channels 28-31 are configured as drivers. The driver data behind the control channels (28-31) is filled with data that will enable one byte at a time to drive to the EPIO. After each byte has been allowed to drive for one vector (for a total of four vectors), all of the bytes will be disabled from driving to the EPIO for four vectors. This eight vector pattern is repeated for the full memory depth. This pattern is used to just vary the disabling of the drivers to the EPIO.

It is highly recommended that this test only be executed when the Selftest Executive is configured to only output fail data. This is because the test performs millions of operations, and if all test data is output to the screen (or file) it would take hours to execute instead of minutes.

Sample Output

```
-------------------------------------------------------------------------------------------------------------
ADDR      0       TST: 1          ADR:Ox6D      EXP:Ox7FFFF    ACT:Ox7FFFF
NML       0       TST: 2          ADR:Ox0       EXP:OxFFFFF00  ACT:OxFFFFF00
RES       0       TST: 3          ADR:Ox0       EXP:OxFFFFF67  ACT:OxFFFFF67
-------------------------------------------------------------------------------------------------------------
```

'ADDR' -> Verify the correct ending address for this test.

'NML' -> Verify correct data was written to the NML Result memory. Any channel on the EPIO that is not driven to will store a logic one to memory. Any channel that is driven to will store a logic zero to memory. In this example, EPIO channels 0-7 were being driven to and the others detected the NML condition.

'RES' -> Verify correct data was written to the Result memory. Any channel on the EPIO that is not driven to will store a logic one to memory. Any channel that is driven to will store the data being driven from the selftest. In this example, EPIO channels 0 - 7 were being driven to and the data was 0x67, the other channels detected the NML condition.

- **If any failures occur during this routine, the problem is most likely on the EPIO board.**
- **Since the Selftest Assembly is required for this test, it could also cause the failures.**
- **Since data is being transfered through the Patchboard, there could be a Patchboard pin not making contact with the Selftest Assembly pin.**
- **Power to the Testhead and connections should always be checked.**

FPS

fps_mon_f

This test checks all of the output voltages of the Fixed Supply Box (FSB) power supplies. The fixed supply outputs are routed to the AMS for measurement via the Patchboard P/S readback register on the Product P/S Selftest board. The AMS should be calibrated before running this selftest. The supply output measurements must be within +/-5% of the rated output voltage to pass the selftest.

- **If failures occur, the problem is most likely in the Fixed Power Supply box.**
- **Since the Amplitude Measurement System is being used to measure the voltages, it could cause problems.**
- **The Selftest Assembly is being used as a signal path and could be faulty.**
- **Power to the Testhead and connections should always be checked.**

ICAM

ADMeas_f

The ADMeas_f selftest program tests the full input range of the analog-to-digital convertor (ADC) on the ICAM board using its test digital-to-analog convertor (TDAC) input channel. The TDAC's output range is tested from -4V to +4V in 0.2V increments. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it initializes the ADC to take readings of its TDAC input channel. The TDAC is programmed to -4.0V before triggering the ADC. The ADC's reading is then retrieved and compared against the minimum and maximum limits. The TDAC is programmed to the next test value, and the trigger-measurement-comparison process is repeated until the entire input range of the ADC has been tested.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**
- **Power to the Testhead and connections should always be checked.**

ADMem_f

The ADMem_f selftest program tests the memory that the analog-to-digital convertor (ADC) on the ICAM board uses to store its readings. It tests the address and data line interfaces as well as all of the locations in the memory chips. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then tests for short-circuited and open-circuited data and address lines. Next, it tests each and every address location by first "marching" a logic 1 followed by a logic 0. Finally, the program tests the hardware that controls the address lines to verify that all of the memory locations are accessible.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Power to the Testhead and connections should always be checked.**

ArbBrst_f

The ArbBrst_f selftest program tests the burst mode of the ARB circuitry on the ICAM board. The complete range of bursts from 1 to 255 is verified. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then initializes the ARB to toggle its MPULSE signal once per cycle and sets up the Time Measurement Circuit (TMC) on the ICAM board to record the number of pulses on its MPULSE input channel within a 100-mS time interval. Next, the ARB's burst counter is configured for 1 pulse before triggering the ARB and the TMC. The TMC's reading is then retrieved and compared to the expected value. The burst counter is incremented to the next test value, and the trigger-measurement-comparison process is repeated until all 255 burst counter settings are tested.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Power to the Testhead and connections should always be checked.**

ArbFreq_f

The ArbFreq_f selftest program tests the frequency accuracy of the ARB's frequency generator on the ICAM board. The test frequencies range from 10 Hz to 10 MHz incrementing on a per-decade basis. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then initializes the TMC to record the number of pulses on its MCLK input channel within either a 10mS or 100mS time interval. Next, the ARB's frequency generator is programmed to a value of 10 Hz before triggering the ARB and the TMC. The TMC's reading is then retrieved and compared against the minimum

and maximum limits. The ARB's frequency generator is reprogrammed to the next incremental (by decade) value, and the trigger-measurement-comparison process is repeated until all of the frequency settings have been tested.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Power to the Testhead and connections should always be checked.**

Arb_I_f

The Arb_I_f selftest program tests the 16-bit ARB circuitry used to generate the 5mA and 50mA bipolar current ranges for the current source on the ICAM board. The 5mA range is tested from -4.5uA to -4.5mA and from +4.5uA to +4.5mA. The 50mA range is tested from -5mA to -50mA and from +5mA to +50mA. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it enables the load resistor on the Turbo Selftest Assembly for the current source's 5mA range. It also sets up the analog-to-digital convertor (ADC) to read the voltage drop across the current source's sense resistor. Then, it programs the ARB to source 4.5uA before triggering the ARB and the ADC. The ADC's reading is then retrieved and compared to the minimum and maximum limits. The ARB is programmed to the next incremental value, and the trigger-measurement-comparison process is repeated until all of the positive and negative 5mA and 50mA current ranges of the ARB are tested.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**
- **Power to the Testhead and connections should always be checked.**

Arb_V_f

The Arb_V_f selftest program tests the 16-bit ARB's output amplifier circuitry and the VOUT signal path to the ICAM measurement circuitry. The ARB is programmed to output a range of voltages from its maximum value to its minimum value. This voltage is then measured using the VOUT signal path to the A/D Converter. The test is performed for both a normal ARB voltage source and a Kelvin voltage source.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Power to the Testhead and connections should always be checked.**

ArbILim_f

The ArbILim_f selftest program tests the current-limiting protection circuitry for the ARB on the ICAM board. It checks both the 200mA and 1A current ranges. This program verifies that the current-limiting protection circuitry is either enabled or disabled under the appropriate circumstances. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it enables the load resistor on the Turbo Selftest Assembly for the ARB. It also sets up the analog-to-digital convertor (ADC) to read the ARB's current output. Then, it enables the ARB's 200mA current range and programs the ARB to source +500mA before triggering the ARB and the ADC. The ADC's reading is then retrieved and compared to the minimum and maximum limits of 200mA and 275mA, respectively. Next, the ARB is programmed to source +175mA, and the trigger-measurement-comparison process is repeated. The ADC's reading is now compared to the nominal value of (175mA multiplied by the load resistance) +/-5%. This is then repeated for the current sinking values of 175mA and 500mA. Next, the ARB's 1A current range is enabled and the ARB's current-sourcing and current-sinking capabilities are tested at the settings of 500mA and 750mA.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s). It may only need to be recalibrated.**

- **Power to the Testhead and connections should always be checked.**

ArbIMon_f

The ArbIMon_f selftest program tests the ARB's current output measurement circuitry. The ARB's 1A current range is tested from -900mA to +900mA in 60mA increments. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it enables the load resistor on the Turbo Selftest Assembly for the ARB. It also sets up the analog-to-digital convertor (ADC) to read the ARB's current output. Then, it enables the ARB's 1A current range and programs the ARB to deliver +900mA before triggering the ARB and the ADC. Next, the ADC's reading is retrieved and compared to the minimum and maximum limits. The ARB is then programmed to the next incremental value, and the trigger-measurement-comparison process is repeated until the entire range is tested.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s). It may only need to be recalibrated.**

- **Power to the Testhead and connections should always be checked.**

ArbMem_f

The ArbMem_f selftest program tests the memory that the ARB on the ICAM board uses to retrieve its wave forms. It tests the address and data line interfaces as well as all of the locations in the memory chips. The Turbo

Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then tests for short-circuited and open-circuited data and address lines. Finally, it tests each and every address location by first "marching" a logic 1 followed by a logic 0.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Power to the Testhead and connections should always be checked.**

Arly_Dig_f

The Arly_Dig_f selftest program tests the digital circuitry on the Auxiliary Relay, Auxiliary FET, Power Relay, and High Current FET boards in the Testhead. The Turbo Selftest Assembly is not used in this functional selftest program.

If multiple boards or a combination of boards exists in the system, the program tests each one in the order indicated by its dip switch settings i.e., 0, 1, 2, 3.

- **If any failures occur, the problem is most likely on the indicated board.**
- **Power to the Testhead and connections should always be checked.**

DCI_f

The DCI_f selftest program tests the secondary 12 bit digital-to-analog convertor (DAC) used to generate the 5mA and 50mA unipolar current ranges for the current source on the ICAM board. The 5mA range is tested from +45uA to +4.5mA. The 50mA range is tested from +5mA to +50mA. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it enables the load resistor on the Turbo Selftest Assembly for the current source's 5mA range. It also sets up the analog-to-digital convertor (ADC) to read the

voltage drop across the current source's sense resistor. Then, it programs the secondary 12-bit DAC to source 45uA before triggering the ARB and the ADC. The ADC's reading is then retrieved and compared to the minimum and maximum limits. The DAC is programmed to the next incremental value, and the trigger-measurement-comparison process is repeated until the 5mA and 50mA current ranges are fully tested.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**
- **Power to the Testhead and connections should always be checked.**

DelayC_f

The DelayC_f selftest program tests the delay counter for the analog-to-digital convertor (ADC) on the ICAM board. Due to time constraints, only 16 tests are performed to verify the integrity of the delay counter hardware. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. Then, it sets the identical frequency for the sample clock and for the ARB. Next, it initializes the ADC's delay counter to the first test delay value. It also programs the ARB to generate a single, unit-long pulse at the same moment that the delay counter will time out and cause the analog-to-digital convertor (ADC) to measure its input. After the ARB and the ADC have been triggered, the ADC's reading of the ARB's output is retrieved and compared to the expected value of 2.5V +/-5%. The delay counter and the ARB are then programmed with the next test value, and the trigger-measurement-comparison process is repeated until all of the test delay values have been checked.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Power to the Testhead and connections should always be checked.**

Event_f

The Event_f selftest program tests the five fixed-frequency clock sources of the ADC's delay counter and the five time interval settings for the Time Measurement Circuit (TMC) on the ICAM board. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. Next, it routes the first fixed-frequency clock source for the ADC's delay counter to the TMC, which is then set up to use its first time interval setting. After triggering, the TMC's pulse count is retrieved and compared against minimum and maximum limits. The next clock source is selected, and the trigger-measurement-comparison process is repeated until all of the possible clock source/time interval combinations have been tested.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Power to the Testhead and connections should always be checked.**

Fixture_f

The Fixture_f selftest program tests the digital circuitry on the ICAM board that interfaces with the fixture. This includes the status lines for the left and right vacuum wells, the fixture ID interface, and the four open-collector driver outputs. The Turbo Selftest Assembly is required for this functional selftest program.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**
- **Power to the Testhead and connections should always be checked.**

IArly_f

The IArly_f selftest program tests the relays (or switches) on all of the Auxiliary Relay, Auxiliary FET, and Power Relay boards in the system. It is assumed that the measurement system is working although it doesn't need to be calibrated. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the Amplitude Measurement System (AMS) board. If the AMS board is present, a separate procedure called nArly_f is executed to test these relay board(s) using the AMS and Relay Multiplexer boards along with the Turbo Selftest Assembly. Otherwise, the program searches the Testhead for the presence of the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it searches for all existing Auxiliary Relay, Auxiliary FET, and Power Relay board and verifies that each board has a corresponding Selftest board.

The reference voltage (generated by the high-precision test digital-to-analog convertor - TDAC, in the Turbo Selftest Assembly) is then applied to the first relay (or switch) on the first board through a resistor. A second, matched resistor connects the other end of the relay (or switch) to ground. The analog-to-digital convertor (ADC) on the ICAM board should measure between 45-55% of the reference voltage across the relay (or switch) when it is closed. The ADC should measure less than 5% of the reference voltage when the relay (or switch) is open.

Then, the rest of the relays (or switches) on the board are checked. If multiple boards or a combination of boards exists in the system, the program tests each one in the order indicated by its dip switch settings i.e., 0, 1, 2, 3.

- **If any failures occur, the problem is most likely on the indicated relay board.**
- **If the AMS board is present, it and/or the Relay Multiplexer board could also cause the failure(s).**
- **If the AMS board is not present, the ICAM board could also cause the failure(s).**

- **Since the Turbo Selftest Assembly is required for this test, it too could cause the failure(s).**

- **Power to the Testhead and connections should always be checked.**

IC_dig_f

This program tests the digital circuitry on the ICAM board. It writes to and reads back from each register. It reports a failure each time that the received data does not match the written data. The Turbo Selftest Assembly is not used in this program.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Power to the Testhead and connections should always be checked.**

IMrly_f

The IMrly_f selftest program tests the relays on all of the Matrix Relay boards in the system. It is assumed that the measurement system is working although it doesn't need to be calibrated. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the Amplitude Measurement System (AMS) board. If the AMS board is present, a separate procedure called nMrly_f is executed to test the Matrix Relay board(s) using the AMS and Relay Multiplexer boards and the Turbo Selftest Assembly. Otherwise, the program searches the Testhead for the presence of the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it searches for all existing Matrix Relay boards and verifies that each board has a corresponding selftest board.

The reference voltage (generated by the high-precision test digital-to-analog convertor - TDAC, in the Turbo Selftest Assembly) is then applied to the first relay on the first board through a resistor. A second, matched resistor connects the other end of the relay to ground. The analog-to-digital convertor (ADC) on the ICAM board should measure between 45-55% of the reference voltage

across the relay when it is closed. The ADC should measure less than 5% of the reference voltage when the relay is open.

Then, the rest of the relays on the board are checked. If multiple boards exist in the system, the program tests each one in the order indicated by its dip switch settings i.e., 0, 1, 2, 3.

- **If any failures occur, the problem is most likely on the indicated Matrix Relay board.**
- **If the AMS board is present, it and/or the Relay Multiplexer board could also cause the failure(s).**
- **If the AMS board is not present, the ICAM board could also cause the failure(s).**
- **Since the Turbo Selftest Assembly is required for this test, it too could cause the failure(s).**
- **Power to the Testhead and connections should always be checked.**

IPower_Mon_f

The IPower_Mon_f selftest program checks the DC voltage outputs of all of the internal Testhead and external Patchboard power supplies. The Testhead board in the first slot must contain a high-voltage test multiplexer (TMUX). The measurement system must have been successfully calibrated. Also, the Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the Amplitude Measurement System (AMS) board. If the AMS board is present, a separate procedure called Power_Mon_f is executed to test the power supplies using the Turbo Selftest Assembly.

Otherwise, the program searches the Testhead for the presence of the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it routes the output of each power supply through the high-voltage TMUX on the first Testhead board to the SIG3 input channel of the analog-to-

digital convertor (ADC) on the ICAM board. A measurement is taken and then compared against the corresponding minimum and maximum limits. The path of the test digital-to-analog convertor (TDAC) in the Turbo Selftest Assembly through the TMUX to the ADC is also tested.

- **If any failures occur, the problem is most likely in the indicated power supply or with its wiring.**
- **If the AMS board is present, it and/or the Relay Multiplexer board could also cause the failure(s).**
- **If the AMS board is not present, the ICAM board could also cause the failure(s).**
- **Since the Turbo Selftest Assembly is required for this test, it too could cause the failure(s).**
- **If multiple power supplies exhibit failures, check the Testhead Power Supply Controller board and the high-voltage TMUX on the Testhead board in the first slot.**
- **Power to the Testhead and connections should always be checked.**

IRail_f

The IRail_f selftest program tests the amplifier circuitry of the "RAILDAC" digital-to-analog convertor (DAC) which provides the adjustable power supply voltage for the current source on the ICAM board. The DAC's output is tested from 3V to 35V in 1.1V increments. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it sets up the analog-to-digital convertor (ADC) to read the voltage output of the current source, VMON. Then, it programs the IRAIL DAC to 3.0V before triggering the ARB and the ADC. Next, the ADC's reading is retrieved and compared to the minimum and maximum limits. The IRAIL DAC is then programmed to the next incremental value, and the trigger-measurement-

comparison process is repeated until the entire output range has been tested.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**
- **Power to the Testhead and connections should always be checked.**

IRDACs_f

The IRDACs_f selftest program tests the "IDAC" and "RAILDAC" digital-to-analog convertors (DACs) on the ICAM board. Each DAC's output is tested from 0V to 40% of full-scale output. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. Next, it sets up the analog-to-digital convertor (ADC) to read its "RAILDAC" DAC's input channel. Then, it programs the "RAILDAC" DAC to -10V before triggering the ARB and the ADC. Next, the ADC's reading is retrieved and compared to the minimum and maximum limits. The "RAILDAC" DAC is programmed to the next incremental value, and the trigger-measurement-comparison process is repeated until the test range has been covered. Then, the ADC is set up to read its "IDAC" DAC's input channel, and the "IDAC" DAC is tested in the same manner as its counterpart.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Power to the Testhead and connections should always be checked.**

Meas_F_f

The Meas_F_f selftest program tests the input frequency range of the Time Measurement Circuit (TMC) on the ICAM board. The test frequencies range from 10 Hz to 3 MHz incrementing on a per-decade basis. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then initializes the TMC to record the number of pulses on its MCLK input channel within either a 10mS or 100mS time interval. Next, the ARB's frequency generator is programmed to a value of 10 Hz before triggering the ARB and the TMC. The TMC's reading is then retrieved and compared against the minimum and maximum limits. The ARB's frequency generator is reprogrammed to the next incremental (by decade) value, and the trigger-measurement-comparison process is repeated until all of the frequency settings have been tested.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Power to the Testhead and connections should always be checked.**

Meas_I_f

The Meas_I_f selftest program tests the current measurement amplifier on the ICAM board. It checks all six current ranges (100mA, 10mA, 1mA, 100uA, 10uA, & 1uA) and their optional Kelvin-measurement connections. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, the analog-to-digital convertor (ADC) is set up to read the output of the current measurement amplifier, VRMEAS. Then, the program selects the first current range of the current measurement amplifier and its corresponding input resistor on the Turbo Selftest Assembly. Next, it programs the test digital-to-analog convertor (TDAC) on the Turbo Selftest Assembly to -8.7V before triggering the ARB and the ADC. Then, the ADC's reading is retrieved and compared to the minimum and maximum limits. The TDAC is programmed to the next incremental value, and the trigger-measurement-comparison process is repeated for the full range of TDAC's output and for the remaining five current ranges.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**

- **Power to the Testhead and connections should always be checked.**

Meas_V_f

The Meas_V_f selftest program tests the four voltage ranges (0.2V, 2V, 20V, & 200V) of each of the four voltage input relay multiplexer (RMUX) groups on the ICAM board. Each group provides a separate input signal to the analog-to-digital convertor (ADC). The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it configures the first RMUX to its 200V range. Then, it routes the output of the test digital-to-analog convertor (TDAC) on the Turbo Selftest Assembly to the first RMUX group. Next, the program sets up the ADC to read the output of the first RMUX group, VMEAS0. Then, it programs the TDAC to -9.375 before triggering the ADC. Next, the ADC's reading is retrieved and compared to the minimum and maximum limits. The TDAC is then programmed to the next incremental value, and the trigger-measurement-comparison process is repeated until the entire 200V range has been tested. Then, the 20V, 2V, and 0.2V ranges are tested in order using the initial TDAC setting of 9.375V, 1.75V, and 0.175V, respectively. Afterwards, the other three RMUX groups are tested in a similar fashion.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**

- **Power to the Testhead and connections should always be checked.**

MRLY_dig_f

This program tests all of the I/O bits that can be read back on the all of the Matrix Relay boards in the system. It also tests the image of the 256 relays in the controller memory, the relay drivers, and the readback serial shift registers on each board.

The second part of the test is accomplished by changing one bit at a time in the image and downloading the image through the driver shift registers. The driver outputs are read back through the readback serial shift registers. Finally, the received values are compared to the set bits, and any failures are recorded. A Turbo Selftest Assembly is required for this test.

- **If any failures occur, the problem is most likely on the indicated Matrix Relay board .**
- **Since the Turbo Selftest Assembly is required for the test, it could also cause the failure(s).**
- **Power to the Testhead and connections should always be checked.**

SampleC_f

The SampleC_f selftest program tests the sample counter of the analog-to-digital convertor (ADC) on the ICAM board. The entire sampling range from 1 to 32767 is tested using the ARB as the input source. The Turbo Selftest Assembly is not used in this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then programs the ARB to a non-zero DC output and routes the signal to the ADC. For each test sample value, the ADC's sample memory is first reset to zero, the ADC's sample counter is programmed with the test sample value, and both the ARB and the ADC are triggered. Once the ADC has completed taking its readings, its memory is checked to verify it contains only the specified number of readings.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Power to the Testhead and connections should always be checked.**

SHFMux_f

The SHFMux_f selftest program tests all of the possible input signal paths (and their relays) involving the four voltage relay multiplexer (RMUX) groups (#0-3) and the current measurement amplifier. Each group has four source bus input channels (#0-3), two high-frequency input channels (#0-1), and a default source bus input source (except for Group #0). The current measurement amplifier can be connected to any one of the twenty-three (23) possible input signals. The Turbo Selftest Assembly is required for this functional selftest program.

The program begins by first searching the Testhead for the ICAM board. It then verifies that the Turbo Selftest Assembly is present and that the assembly contains an ICAM Selftest board in the same slot as the ICAM board. Next, it programs the test digital-to-analog converter (TDAC) on the Turbo Selftest Assembly , the ARB, and the secondary 12-bit current source DAC to provide the test signals that will be routed through the various signal paths and relays. Then, the program configures the first RMUX to select its four source bus input channels and two high-frequency input channels one at a time in order whereby the analog-to-digital converter (ADC) is triggered for each configuration. The ADC's reading is retrieved and compared to the minimum and maximum limits. Next, the input paths and relays of the remaining three groups are tested in a similar manner. Also, the output of the secondary 12-bit current source DAC is routed to the current measurement amplifier and tested to verify its input paths.

- **If any failures occur, the problem is most likely on the ICAM board.**

- **Since the Turbo Selftest Assembly is required for this test, it could also cause the failure(s).**

- **Power to the Testhead and connections should always be checked.**

XO_Phase_f

The XO_Phase_f selftest program tests the crossover detectors on the ICAM board. This circuit can be used in conjunction with the Time Measurement Circuit (TMC) to measure the frequency of the ARB's output waveform. The Turbo Selftest Assembly is not used in this functional selftest program.

- **If any failures occur, the problem is most likely on the ICAM board.**
- **Power to the Testhead and connections should always be checked.**

INST/ISO

AMP_cmrr_f (Inst_cmrr_f)

This selftest program measures and checks the Common Mode Rejection Ratio of the four Inst/Iso amplifiers installed on an Instrumentation Amplifier board, Isolation Amplifier board, or a MSP board. This test requires a Selftest Assembly, a Relay Multiplexer board, and an Amplitude Measurement System board.

The differential inputs are connected together at a 0.0 volt potential, and a measurement of the amplifier output is taken for gain and offset corrections. Each amplifier is then fed +9 volts of common mode voltage input, using TDAC as a source, and measurements are taken. The Common Mode Rejection Ratio is calculated based on these readings.

- **If the test fails, the Instrumentation Amplifier board is most likely at fault.**
- **Since the Selftest Assembly, a Relay Multiplexer board, and the Amplitude Measurement System are also used during the test, they should also be checked.**
- **Power to the Testhead and connections should always be checked.**

IAmp_dig_f (Inst_dig_f)

This program is designed to test the digital circuitry of all instrumentation or isolation amplifiers on all types of boards in the system. A Selftest Assembly is necessary for this test.

When the test is run, the program first determines if a Selftest Assembly is present, and the type of Selftest if one exists. The program then searches the Testhead for an Instrument Amplifier board, an Isolation Amplifier board, or a Multiple Serial Protocol board and IDs it. The latches that control the filter and gain circuitry for the board are written to and read back. If the data doesn't match, a failure is flagged.

Next, the Patchboard ID circuitry is tested by walking a 1 through a register on the Selftest Assembly which is connected to the Patchboard ID pins of the board in slot 1.

- **If the read/write operation on the latches fails, the Instrumentation Amplifier board, Isolation Amplifier board, or Multiple Serial Protocol board should be replaced.**

- **If the Selftest ID portion of the test fails, the Instrumentation Amp, Isolation Amp, or Multiple Serial Protocol board should be checked.**

- **Power to the Testhead and connections should always be checked.**

Inst_f

This program tests any instrumentation amplifier found in the system. There can be a maximum of one Instrumentation Amplifier board (containing four amplifiers) in the Testhead. Each channel is tested over its specified output range of +/-10V by measuring output through the Amplitude Measurement System board through Sig3.

Initially the program checks for the presence of a Selftest Assembly and determines if the Selftest is a standard or Turbo type. Next, the program searches the Testhead for the Analog Source board. The program then searches the Testhead for an Instrumentation Amp board. If a Turbo Selftest is present, the TDAC is used to provide voltages for the Instrumentation Amps. If not, the D/As on the Analog Source board provide the three voltages to be tested, -10V, 0V, and +10V. The output of the Instrumentation Amp is routed to the TMUX. The output from the TMUX (Sig3) is measured by the Amplitude Measurement System board. The gain stages and complete voltage ranges of the amplifiers are tested for all four channels.

- **If a failure occurs, the problem is most likely in the Instrumentation Amplifier board.**

- **Since the Analog Source board, Amplitude Measurement System, TMUX, and Selftest Assembly are also used in this test, they should also be checked.**

- **Power to the Testhead and connections should always be checked.**

IsoAmp_f

This program tests any isolation amplifier found in the system. There can be a maximum of one board (Isolation Amplifier, or Multiple Serial Protocol) containing four amplifiers in the Testhead. Each channel is tested over its specified output range of +/-10V by measuring output through the Amplitude Measurement System board through Sig3.

Initially the program checks for the presence of a Selftest Assembly and determines if the Selftest is a standard or Turbo type. Next, the program searches the Testhead for the Analog Source board, and then searches the Testhead for an Isolaton Amplifier board. (For a Standard Selftest Assembly, the program only searches for an Isolation Amplifier). If a Turbo Selftest is present, the TDAC is used to provide voltages for the Instrumentation Amps. If not, the D/As on the Analog Source board provide the three voltages to be tested, -10V, 0V, and +10V. The output of the Instrumentation Amp is routed to the TMUX. The output from the TMUX (Sig3) is measured by the Amplitude Measurement System board. The gain stages and complete voltage ranges of the amplifiers are tested for all four channels.

- **If a failure occurs, the problem is most likely in the Isolation amplifier or Multiple Serial Protocol board.**
- **Since the Analog Source board, Amplitude Measurement System, TMUX, and Selftest Assembly are also used in this test, they should also be checked.**
- **Power to the Testhead and connections should always be checked.**

IVAL

Ival_Dig_f

The Ival_dig_f program tests the digital circuitry on the IVAL board. It writes to and reads back from each register. It reports a failure each time that the received data does not match the written data. The Turbo Selftest Assembly is not used in this program.

- **If any failures occur, the problem is most likely on the IVAL board.**
- **Power to the Testhead and connections should always be checked.**

MDE

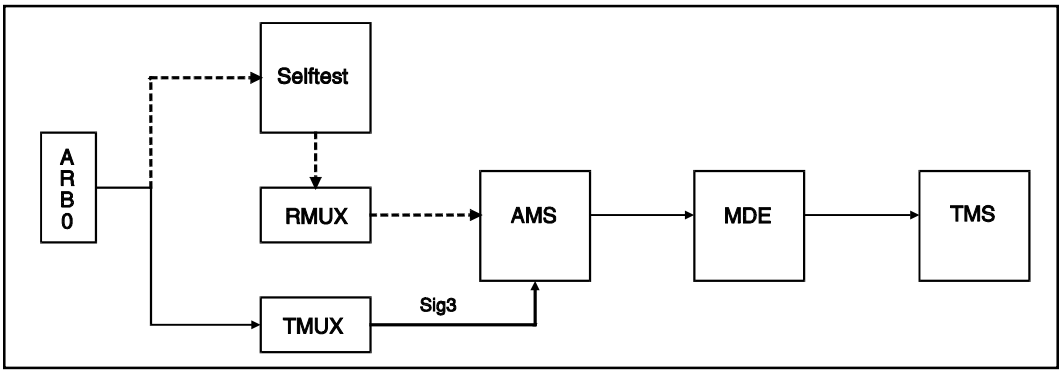
MDE_delmodes_f

The MDE_delmodes_f program tests the Trace2 delay modes, and requires the use of the Time Measurement System, Amplitude Measurement System, Measurement Display Electronics, and Analog Source board. This program does not require a Selftest Assembly.

Initially the program searches the Testhead for the Analog Source board, Amplitude Measurement System, and Measurement Display Electronics boards. The Analog Source board, Amplitude Measurement System, and Measurement Display Electronics are then set up to generate and gate a specific waveform while the Time Measurement System checks the sweep time. Trace2 is started at various delay points relative to Trace1, and the delays are checked by the Time Measurement System board.

The normal signal path for this test (without a Selftest Assembly installed) is through the TMUX to the Amplitude Measurement System via Sig3. If a Selftest is installed, the program uses a path through the Selftest Assembly and a Relay Multiplexer channel to the Amplitude Measurement System, as shown in the block diagram below.

- **If errors occur, the Measurement Display Electronics board is usually at fault.**
- **The Analog Source board, TMUX, Amplitude Measurement System, and Time Measurement System are also used in the test and should also be checked.**
- **Power to the Testhead and connections should always be checked.**



MDE_deltime_f

This program exercises the delay circuitry of the Measurement Display Electronics board and checks the delay calibration. It also exercises the gates that allow time measurements from the start of the trace to the end of the delay. This program requires a Selftest Assembly.

Initially the program checks for the presence of a Selftest Assembly, and determines if the Selftest is a standard or Turbo type. Then the presence of an Analog Source board and Measurement Display Electronics board in the Testhead is determined. ARB0 is programmed with a specific waveform, and the Measurement Display Electronics is set up. Several tests are performed with various delay and sweep times while the Time Measurement System board measures the results.

The path for the waveform is from ARB0 through the Selftest Assembly, through a Relay Multiplexer board, through the Amplitude Measurement System, through the Measurement Display Electronics, and ends at the Time Measurement System.

- **If failures occur, the Measurement Display Electronics board is usually at fault.**
- **The Analog Source board, Relay Multiplexer, Amplitude Measurement System, and Time Measurement System are also used during the test and should be checked.**
- **Since a path through the Selftest Assembly is used in this test, the Selftest Assembly should also be checked.**
- **Power to the Testhead and connections should always be checked.**

MDE_extpmeas_f

The MDE_extpmeas_f program tests the extended period measurement gates numbers 1 and 15. It will, however, test only Measurement Display Electronics boards with ECO #1 revisions on the board. No Selftest Assembly is required for this test.

The program searches the Testhead for the Analog Source and Measurement Display Electronics boards. When this is done, ARB0 is programmed to generate a squarewave. A second waveform is programmed for Trace2, and triggers are set. The Time Measurement System is used to measure certain triggers on the waveforms and compare the readings to preset values.

- **If a failure occurs, the problem is usually on the Measurement Display Electronics board.**
- **Since the Analog Source board, TMUX, Amplitude Measurement System and Time Measurement System boards are also used in the test, they should also be checked.**
- **Power to the Testhead and connections should always be checked.**

MDE_freq_f

The MDE_freq_f program exercises the crossover detectors and compares each test frequency to the frequency that ARB0 was programmed to. This test does not require a Selftest Assembly.

Initially the program searches the Testhead for the Analog Source and Measurement Display Electronics boards. ARB0 on the first Analog Source board generates a squarewave, which is sent through the TMUX and Amplitude Measurement System board to the Measurement Display Electronics. The squarewave is then gated by the Measurement Display Electronics and the frequency is measured by the Time Measurement System board. All three triggers, TrigA, Trig1, and Trig2 are exercised during this test.

- **If the test fails, the Measurement Display Electronics board is usually at fault.**
- **The Analog Source board, TMUX, Amplitude Measurement System, and Time Measurement System boards are also used during the test and should also be checked.**
- **Power to the Testhead and connections should always be checked.**

MDE_markplace_f

The MDE_markplace_f program exercises the measurement mark and its ability to be moved to specific positions on either trace displayed by the Measurement Display Electronics. In doing so, it tests the mark calibration and the circuitry that allows time measurements to be made from the beginning of the sweep to mark.

Initially the program checks for the presence of a Selftest Assembly, and determines if the Selftest is a standard or Turbo type. Then the program searches the Testhead for an Analog Source board and a Measurement Display Electronics board. ARB0 is programmed to generate a squarewave and the waveform and sweep times are checked by the Time Measurement System. The mark command is run, and the program uses the Time Measurement System to measure and validate the position of mark on the waveform.

The path for the waveform is from ARB0 through the Selftest Assembly, through a Relay Multiplexer board, through the Amplitude Measurement System, through the Measurement Display Electronics, and ends at the Time Measurement System.

- **Any failures in this test are usually caused by a faulty Measurement Display Electronics board.**
- **Since the Analog Source board, Relay Multiplexer, Amplitude Measurement System, and Time Measurement System boards are also used during the test, they should also be checked.**
- **Since a path through the Selftest Assembly is used in this test, the Selftest Assembly should also be checked.**
- **Power to the Testhead and connections should always be checked.**

MDE_measmark_f

The MDE_measmark_f program checks the interface of the measurement mark of the Measurement Display Electronics to the trigger circuit on the Amplitude Measurement System board. The system must be calibrated to run this test. The program also checks the interface of the measurement mark to the Time

Measurement System. A Selftest Assembly is not required for this test.

The program first searches the Testhead for the Analog Source and Measurement Display Electronics boards. ARB0 is programmed to generate a squarewave, while the gate times and mark voltages are checked against limits.

- **If failures occur, the problem is usually on the Measurement Display Electronics board.**
- **Since the Analog Source board, TMUX, Amplitude Measurement System, and Time Measurement System boards are also used by this test, they should also be checked.**
- **Power to the Testhead and connections should always be checked.**

MDE_permeas_f

The MDE_permeas_f program exercises the ability to gate signals out to the Time Measurement System board from the Measurement Display Electronics. This program does not require a Selftest Assembly.

Initially the program searches the Testhead for the Analog Source and Measurement Display Electronics boards. The Analog Source board generates a squarewave which is sent through the TMUX and Amplitude Measurement System board to the Measurement Display Electronics. The squarewave is then gated by the Measurement Display Electronics board and the period is measured by the Time Measurement System board. All three triggers, TrigA, Trig1, and Trig2 are exercised during the test.

- **If the test fails, the Measurement Display Electronics board is usually at fault.**
- **The Analog Source board, TMUX, Amplitude Measurement System, and Time Measurement System boards are also used during the test and should be checked.**
- **Power to the Testhead and connections should always be checked.**

MDE_trigfilt_f

The MDE_trigfilt_f program exercises the filters on the crossover detectors. It checks the filters individually by measuring the phase shift of the waveform generated by ARB0. A Selftest Assembly is necessary for this program.

The program first checks for the presence of a Selftest Assembly. It then searches the Testhead for the Analog Source board, Amplitude Measurement System, and Measurement Display Electronics boards. The Analog Source board is programmed to generate a squarewave. The three triggers, TrigA, Trig1, and Trig2, are set up and measured by the Time Measurement System board. The filters are changed, and the Time Measurement System board measures the phase shift in the waveform due to the change in filtering.

The path for the waveform is from ARB0 through the Selftest Assembly, through a Relay Multiplexer board, through the Amplitude Measurement System, through the Measurement Display Electronics, and ends at the Time Measurement System.

- **If failures occur, the Amplitude Measurement System board is usually at fault.**
- **Since the Analog Source board, Relay Multiplexer, Measurement Display Electronics, and Time Measurement System boards are also used in the test, they should also be checked.**
- **Since a path through the Selftest Assembly is used in this test, the Selftest Assembly should also be checked.**
- **Power to the Testhead and connections should always be checked.**

MDE_triglev_f

This program exercises the ability of the crossover detectors to adjust to different trigger levels. This test also exercises the Measurement Display Electronics ability to trigger on different slopes. It uses a triangular waveform from the Analog Source board and the Time Measurement System to accomplish this. This program requires a Selftest Assembly.

The program first checks for the presence of a Selftest Assembly. Then the program searches the Testhead for the Analog Source board, Amplitude Measurement System, and Measurement Display Electronics boards. ARB0 is programmed to generate a triangular waveform. The T2DEL call is used to start a second waveform, and a mark is assigned on the first waveform. TrigA, Trig1, and Trig2 are set up, and the gate times between them are measured by the Time Measurement System. The Measurement Display Electronics is then tested for triggering on rising and falling edges.

The path for the waveform is from ARB0 through the Selftest, through a Relay Multiplexer board, through the Amplitude Measurement System, through the Measurement Display Electronics, and ends at the Time Measurement System.

- **If failures occur, the problem is most likely on the Amplitude Measurement System or Measurement Display Electronics boards.**
- **Since the Relay Multiplexer and Time Measurement System boards are also used in the test, they should also be checked.**
- **Since a path through the Selftest Assembly is used in this test, the Selftest Assembly should also be checked.**
- **Power to the Testhead and connections should always be checked.**

MFRly

mformrly_f

The mformrly_f selftest program tests the relays on all the Multi-Form Relay boards found in the testhead. A Turbo Selftest Assembly and an RMux board are required to run this test. This test also assumes that the Amplitude Measurement System is working although it need not be calibrated.

The TDAC is set to a reference of 6V and is routed to a test relay through a resistance. The Multi-Form Relay under test is connected between the test relay and a matched resistance to ground. The relay is assumed good if the voltage measured at the point between the relay and the resistor is less than 5% of the reference when the relay is open, and 50% (+/- 5%) when the relay is closed. Each of the 44 relays on the Multi-Form board are to be tested one at a time. The test will be repeated with a negative reference voltage.

- **If failures occur, the problem is most likely on the MFRly board.**
- **The Relay Multiplexer and Selftest are being used as signal paths and could be faulty.**
- **Power to the Testhead and connections should always be checked.**

mformrly_dig_f

The mformrly_dig_f selftest program tests the digital logic on all the Multi-Form Relay boards found in the testhead. This program will be split into two tests. The first test will walk a logic '1' through each of the five write registers and verify the data input latches by reading back the corresponding data readback register. The next test will walk a logic '1' through each of the five write registers and verify the relay driver logic by reading back the corresponding relay driver readback register. A Turbo Selftest Assembly is not used during this test.

- **If failures occur, the problem is most likely on the MFRly board.**
- **Power to the Testhead and connections should always be checked.**

Miscellaneous Tests

SELF_dig_f

This program was written to test the digital circuitry in the Turbo Selftest Assembly.

Initially, the program determines the presence of a Selftest Assembly. If the Selftest Assembly is a standard style, the program runs TRLY_dig_f. If the Selftest Assembly is a Turbo style, the program executes SELF_dig_f. Once the Turbo Selftest Assembly is detected and identified, the program writes to and reads from every board in the Selftest Assembly. Next, the test relay latches are written to and read from.

- **If any failures are detected during this test, the Selftest Assembly MUST be sent back to Digalog for repair.**

- **Power to the Testhead and connections should always be checked.**

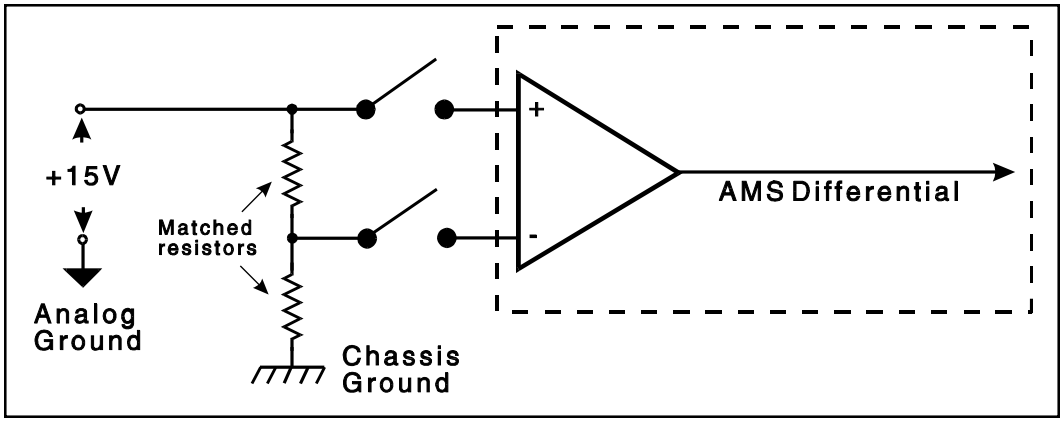
TH_config_tst

The TH_config_tst is a Windows 95™ system resource test which verifies that the present tester configuration matches the configuration setup inscribed in the resource.ini file. The resource.ini file is created and/or modified by executing the Digalog program TRMAN to configure the 2040 tester. Any current changes in the hardware configuration from the configuration specified in the resource.ini file will be flagged.

NOTE: On systems running the Windows NT 4.0 or Windows XP operating system, the resource.ini file is not used. Instead, the tester configuration is stored in the system Registry.

TH_iso_f

This Selftest program checks the isolation of the Measurement System ground from the Testhead chassis. The circuit on the next page indicates the method by which a short is detected between the two grounds.



Two matched resistors are placed in series between +15 volts (referenced to analog ground) and chassis ground. If a dead short exists between the two grounds anywhere in the system, approximately 7.5 volts will be measured by the Amplitude Measurement System during the test. If the voltage measured is less than a predetermined value, the test passes and the Measurement System is considered isolated.

- **Make sure the X-Y-Z coaxial cables are disconnected from the back of the tester since the system oscilloscope may tie the grounds together.**

- **Check for internal shorts in cabling such as the X-Y-Z coaxial cables, power supplies, etc.**

TRLY_dig_f

This program was written to test the digital circuitry in the standard Selftest Assembly.

Initially the program determines the presence of a Selftest Assembly. If the Selftest Assembly is a standard style, the program runs TRLY_dig_f. If the Selftest Assembly is a Turbo style, the program branches to SELF_dig_f. The test relay latches and the QMUX0 and QMUX1 latches are written to and read from.

- **If any failures are detected during this test, the Selftest Assembly MUST be sent back to Digalog for repair.**

- **Power to the Testhead and connections should always be checked.**

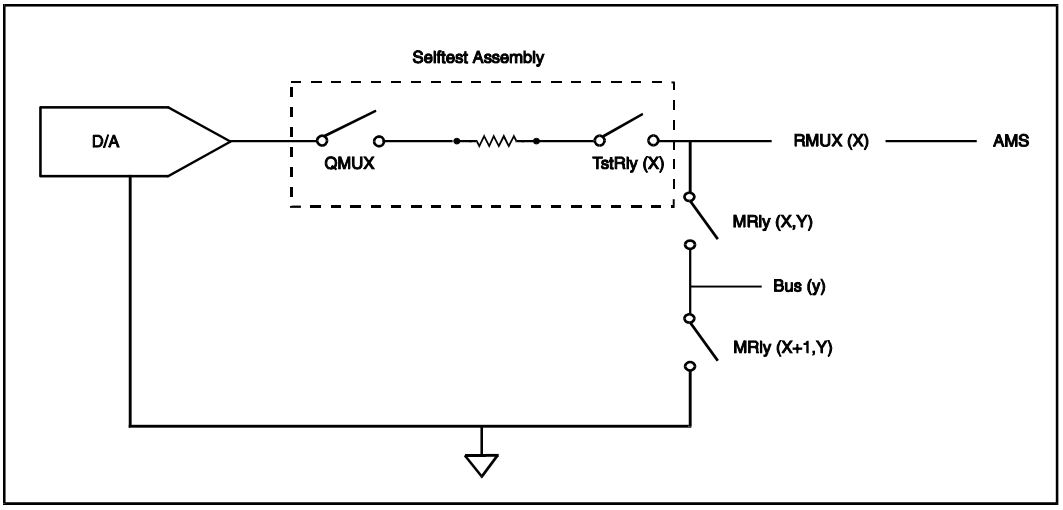
MRly

MRLY_f

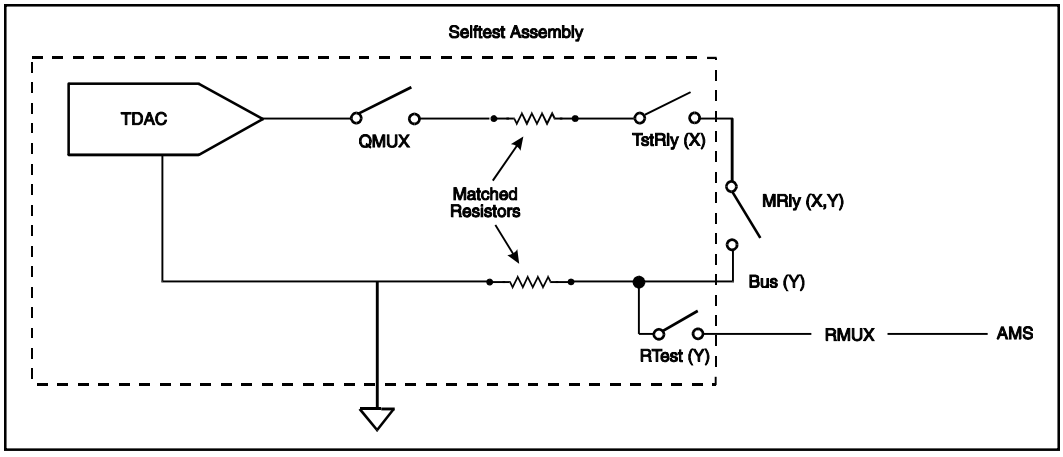
The MRLY_f program exercises the relays on all of the Matrix Relay boards in the system. It requires a Selftest Assembly and a Relay Multiplexer board. (It may also require a D/A channel from an Analog Source board, if a standard Selftest Assembly is used.) It is assumed that the Measurement System is working, although it doesn't need to be calibrated.

The test procedure is done in one of two ways. When a Turbo Selftest Assembly is detected, the MRLY_f program calls a separate procedure (nMRLY_f) to test the Matrix Relay boards in an improved manner, as described below.

When operating with a standard Selftest Assembly, Matrix Relays are tested in pairs. The reference voltage is generated by a D/A and routed through the Selftest Assembly. One channel is connected to the test source (generated by the D/A) and the other is connected to analog ground. In sequence, the two channels are connected and disconnected in every possible combination on all four buses. Using this method, it is possible to determine which physical relay on the board may be at fault by checking the measured voltage at the source point. If either one of the relays is closed, less than 10% of the reference voltage should be measured by the Amplitude Measurement System. If both relays are open, the full reference voltage should be measured.



When operating with a Turbo Selftest Assembly, the procedure is very different. The reference is generated by the Selftest Assembly high-precision TDAC,



making the test independent of the Analog Source board. Also, every relay on the Matrix Relay board may be checked individually, isolating the cause of the failure down to a single relay on a specific board in the Testhead.

This test is again improved by taking measurements at the bus . This is not done using the standard Selftest Assembly, and faults between the bus and Patchboard would not have been detected. A relay is assumed good if the voltage read across the resistor is between 45 - 55% of the reference voltage with the relay closed, and less than 5% with the relay open.

- **If a failure occurs while operating with a standard Selftest Assembly, the problem may be on any Relay or Matrix Relay board in the system. The boards should be removed, one at a time, to isolate the problem to the correct board.**

- **If a failure occurs while operating with a Turbo Selftest Assembly, the problem is likely to be the Matrix Relay board indicated by the error message.**

- **Since the Relay Multiplexer and Amplitude Measurement System boards and the Selftest Assembly are also used in the test, these should also be checked.**

- **Power to the Testhead and connections should always be checked.**

MRLY_dig_f

A Selftest Assembly is required for this test. This program tests all the I/O bits that can be read back. In addition, the program also tests the image of the 256 relays in the controller memory, the relay drivers, and the readback serial shift registers. The second part of the test is accomplished by changing one bit at a time in the image, and downloading the image through the driver shift registers. Then the driver outputs are read back through the readback serial shift registers. Finally, the readback values are compared to the set bits, and any failures are recorded.

- **If errors occur, the Matrix Relay board is most certainly the problem.**
- **Since the Selftest Assembly is used to control some circuitry during the test, it should also be checked.**
- **Power to the Testhead and connections should always be checked.**

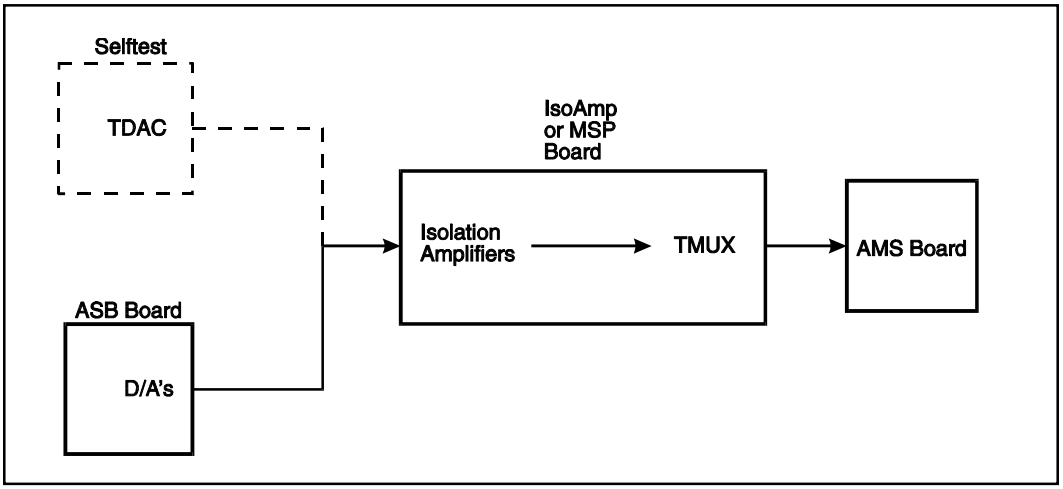
MSP

MSP_amp_f

This program tests the four isolation amplifiers on the Multiple Serial Protocol board. Each of the four channels is tested over its specified output range of +/-10V by measuring the output through the Amplitude Measurement System board through Sig3.

Initially the program checks for the presence of a Selftest Assembly and determines if the Selftest is a standard or Turbo type. Next, the program searches the Testhead for a Multiple Serial Protocol board. If a Turbo Selftest Assembly is present, the TDAC is used to supply voltages for the Isolation Amplifiers. If not, the D/As on the Analog Source board provide the three voltages to be tested, 10V, 0V, and +10V. The output from the Isolation Amplifiers is routed to the TMUX. The output from the TMUX (Sig3) is measured by the Amplitude Measurement System board. The gain stages and complete voltage ranges of the amplifiers are tested for all four channels.

- **If failures occur, the problem is most likely in the amplifier section of the Multiple Serial Protocol board.**
- **Since the Analog Source board, Amplitude Measurement System, TMUX, and Selftest are also used in this test, they should also be checked.**
- **Power to the Testhead and connections should always be checked.**



MSP_dig_f

The MSP_dig_f program merely writes to specific registers on the Multiple Serial Protocol board, and reads back the contents to verify communication. A Selftest Assembly is necessary for this program.

The program initially checks for the presence of a Selftest Assembly, and determines if the Selftest Assembly is a standard or Turbo type. Then the Multiple Serial Protocol board is written to and read from.

Next, the Selftest Assembly is read and identified. If the Selftest Assembly is a standard type, the program reads the ID information. If a Turbo Selftest Assembly is present, the program writes to and reads from every board in the Selftest Assembly.

- **If failures occur in this routine, the problem is almost always in the Multiple Serial Protocol board.**
- **Since the Selftest Assembly is used in this test, it should also be checked.**
- **Power to the Testhead and connections should also be checked.**

MSP_serial_f

The MSP_serial_f program exercises the primary serial ports on the Multiple Serial Protocol board. This test requires a Turbo Selftest Assembly.

Initially the program checks for the presence of a Turbo Selftest Assembly. Next, it checks for the presence of a Multiple Serial Protocol Selftest board in the Selftest Assembly. Then it searches the Testhead for a Multiple Serial Protocol board. When the presence of all three is verified, the program sets up and tests the RS-232 at 8192 baud.

- **If failures occur during this test, the problem is almost always on the Multiple Serial Protocol board.**
- **Since the Turbo Selftest Assembly is used during the test, it should also be checked.**

- **Power to the Testhead and connections should always be checked.**

OCIO

OCIO_f

This routine checks the Driver/Receiver sections of the OCIO board. A Turbo Selftest Assembly is required.

The first portion of this test checks the operation of each channel's driver and receiver using a "walking 1's" test. Each channel's I/O Patchboard pin is connected to a parallel-to-serial shift register on the Selftest board. The data read through the Selftest is compared to the driver data. If it does not match, the driver is assumed to be at fault, or the Patchboard is not making a connection. If the data read through the Selftest does not match the data read by the receivers, then the receivers are assumed to be at fault. To remove the possibility that a channel's pull-up enable circuitry could be causing a fault, the Selftest board must provide external pull-up resistors.

The second portion of the test checks the pull-up circuitry of all OCIO channels, using external pull-down resistors. If the pull-up circuit is not functioning properly then the data read back from the channel will be zero.

- **If a failure occurs, the problem is most likely in the OCIO board itself.**
- **Since the Selftest Assembly is used during the tests, it could also be faulty.**
- **Power to the Testhead and connections should always be checked.**

OCIO_dig_f

This program checks the reset states of the control register, driver enable registers, driver data registers, and the driver data latches. It also performs a bit shift test on the control register and a "walking 1's" test on the eight enable registers on each OCIO board. A "walking 1's" test is similar to the bit shift test; however, every enable register will be read with each shift to insure that data is only being written to a single location.

No digital tests will be performed on the eight receiver latches or eight driver latches, as they are not true read/write registers. The OCIO Selftest board is not used during this test.

- **If a failure occurs, the problem is most likely in the ADIO board itself.**

- **Power to the Testhead and connections should always be checked.**

OCIO_rail_f

This routine checks the external rail circuitry of the OCIO board. The test requires a Turbo Selftest Assembly.

To check the external rail voltage switching circuits, the external rail inputs to each of the two banks must be programmable between +5V and digital ground. Operation of the banks is tested by setting all the OCIO output data to high and external rail to +5V. Data read back should all be high. The external rail is then set to digital ground, and the data read back should now be low.

- **If a failure occurs, the problem is most likely in the ADIO board itself.**

- **Since the Selftest Assembly is used during the tests, it could also be faulty.**

- **Power to the Testhead and connections should always be checked.**

PPS

PSC_dig_f

This Selftest program tests the digital circuitry, the Trigger Matrix, and the functionality of the 'CAPTRGx' signals on the Testhead Power Supply Controller (PSC) board. This test is only valid for 0000-6026 or later PSC boards. A selftest fixture is not needed.

To test the digital circuitry a walking bit test is performed on each of the read/write registers. Each bit is tested individually as well as all bits low.

To test the Trigger Matrix each of the eight outputs are connected to each of the eight inputs. Each of these connections is done on a different bus. A logic high level is given on only one Trigger Matrix output and then its verified that only the one correct Trigger Matrix input received the logic high level.

The 'CAPTRGx' signals are latched Trigger Matrix input signals. One for each of the eight inputs. Each 'CAPTRGx' signal also has a polarity control bit to control if the input should be latched on a logic one level or logic zero level. In the first part of this test the Trigger Matrix is setup so that each output signal is connected to one input signal. Then each output signal is toggled separately. This should toggle only one input which should only clock one 'CAPTRGx'. It is verified that the 'CAPTRGx' signal only gets latched with the correct level change on the Trigger Matrix input signal. The second part of the test checks the ability to reset individual 'CAPTRGx' signals. The Trigger Matrix is again setup to route one output to one input. All of the 'CAPTRGx' lines are latched high by toggling each of the eight outputs. Then one 'CAPTRGx' line is reset at a time and verified.

- **If a failure occurs, the problem is most likely in the PSC board itself.**

- **Power to the Testhead and connections should always be checked.**

POWER_mon_f

The POWER_mon_f program checks all of the voltages of the internal Testhead and external Patchboard power supplies. AC ripple on the supply voltages is also checked. The Amplitude Measurement System board should be in calibration. No Selftest Assembly is required.

After the power supplies are set up, the TMUX and Amplitude Measurement System are programmed. Next, the test is run and checks each supply for proper voltage +/-0.5V and ripple +/-0.1V.

- **If failures occur, it is usually the power supply that failed.**
- **If multiple power supplies fail, check the Testhead Power Supply Controller or the TMUX on the Instrumentation Amplifier, Isolation Amplifier, or Multiple Serial Protocol boards.**
- **Power to the Testhead and connections should always be checked.**

PTEST_f

This program uses the Selftest Assembly to test both the linearity and accuracy of the voltage and current outputs of each of the product power supplies. For each power supply, it first tests the linearity of voltage and current. The programmed steps range from 5% of full scale to 95% of full scale. Next, the program tests the voltage and current accuracy of the power supplies at 25%, 50%, and 75% of the power supplies' full rated output.

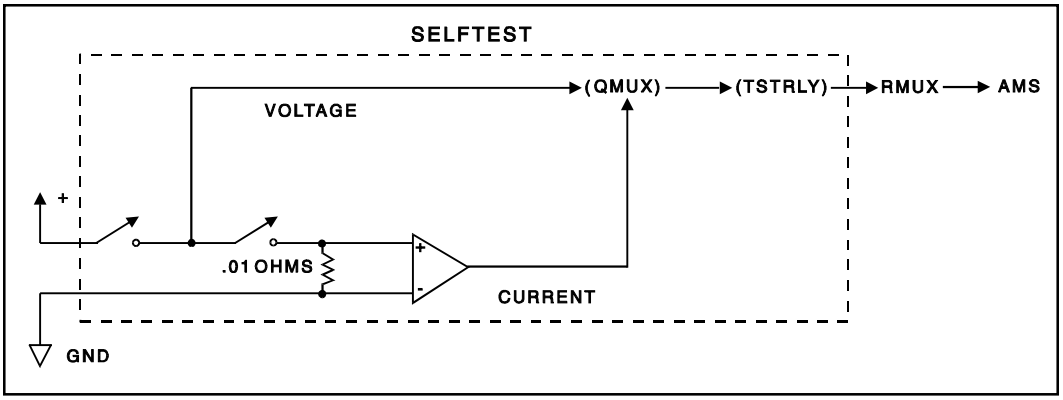
Initially the program checks for the presence of a Selftest Assembly and determines if the Selftest is a standard or Turbo type. If a Turbo Selftest Assembly is detected, an alternate program (nPTest_f) is executed, and has a different setup procedure. The program will then reset and identify each UUT power supply in the system and determine its range. Next, the voltage and current linearity tests are performed. As mentioned above, the supplies are tested for voltage and current accuracy at three points in their full scale range.

For linearity tests:

The programmed voltage/current is measured by the controller itself.

For accuracy tests:

The programmed voltage/current is measured by the Amplitude Measurement System. A block diagram of the Selftest circuitry is shown on the next page.



- **Ensure that the UUT Power Supply is switched on.**
- **If a failure occurs, the UUT Power Supply Controller is usually at fault.**
- **Since the Amplitude Measurement System, Relay Multiplexer, and Selftest Assembly are also used in the test, they should also be checked and calibrated.**
- **If the above suggestions fail to resolve the problem, replace the indicated power supply itself.**

PRO

PrDom_f

PrResO_C must be executed prior to this test. A Turbo Selftest Assembly is required.

The PrDom_F program verifies the correct calibration and operation of the PRO ohmmeter. The certified resistance values located on the selftest board in the opposing slot of the Selftest Assembly are measured by the PRO ohmmeter. The measured resistance values are verified to PRO board specifications. Each of the eight multiplexed inputs is tested in the same way.

- **If failures occur in this routine, the problem is almost always in the Programmable Resistor/Ohmmeter board.**
- **Since the Selftest Assembly is used in this test, it should also be checked.**
- **Power to the Testhead and connections should also be checked.**

PrDvm_f

PrResV_C must be executed prior to this test. A Turbo Selftest Assembly is required.

The PrDvm_F program verifies the correct calibration and operation of the PRO voltmeter. The TDAC is programmed to several values within each range and measured by the PRO voltmeter. The measured voltage values are verified to PRO board specifications. Each of the eight multiplexed inputs is tested in the same way.

- **If failures occur in this routine, the problem is almost always in the Programmable Resistor/Ohmmeter board.**
- **Since the Selftest Assembly is used in this test, it should also be checked.**
- **Power to the Testhead and connections should also be checked.**

PrDvrm_f

PrResV_C must be executed prior to this test. A Turbo Selftest Assembly is required.

The PrDvrm_F program verifies the PRO capability of measuring the voltage across any one (1) of the four (4) programmable resistors within specifications. Several TDAC voltages are applied across each of the four (4) programmable resistors and measured with the PRO voltmeter. The measured voltages are verified to specifications.

- **If failures occur in this routine, the problem is almost always in the Programmable Resistor/Ohmmeter board.**
- **Since the Selftest Assembly is used in this test, it should also be checked.**
- **Power to the Testhead and connections should also be checked.**

PrFloat_f

PrResV_C must be executed prior to this test. A Turbo Selftest Assembly is required.

The PrFloat_F program checks the integrity of the isolation of each of the four (4) programmable resistors and the voltmeter from the rest of the system. The isolation of each of the four (4) programmable resistors is tested by connecting VCC to one side of the programmable resistors being tested through a one-hundred (100) kilo-ohm resistor. The other side of the programmable resistor under test is left unconnected and programmed to a small value to accentuate the quality of the isolation. The voltage is measured at the junction of the one-hundred (100) kilo-ohm resistor and the programmable resistor under test. If the programmable resistor under test is isolated, the voltage drop across the one-hundred (100) kilo-ohm resistor connected in series to VCC will be very small. The voltage measured is verified to specification compared with a voltage measurement of VCC taken before the programmable resistor under test was connected.

To test the voltmeter isolation, VCC is connected to the voltmeter through a one-hundred (100) kilo-ohm resistor and ground is left unconnected. A programmable resistor is placed across the voltmeter to prevent the inputs from floating. The programmable resistor is programmed to a large value to accentuate the isolation characteristic. If the voltmeter is isolated, the voltage measured is verified to a specification close to zero (0) volts.

- **If failures occur in this routine, the problem is almost always in the Programmable Resistor/Ohmmeter board.**
- **Since the Selftest Assembly is used in this test, it should also be checked.**
- **Power to the Testhead and connections should also be checked.**

PrRes_F

PrResO_C must be executed prior to this test. A Turbo Selftest Assembly is required.

The PrRes_F program measures the PRO programmable resistors with the ohmmeter located on the PRO board. Each of the four (4) programmable resistors are programmed to a range of values selected to sufficiently test the accuracy of each resistance on the board. The Selftest Assembly is used to route each of the four (4) programmable resistors back to the PRO Ohmmeter for measurement. The PRO Ohmmeter is then used to validate the programmed resistance to a specified accuracy.

- **If failures occur in this routine, the problem is almost always in the Programmable Resistor/Ohmmeter board.**
- **Since the Selftest Assembly is used in this test, it should also be checked.**
- **Power to the Testhead and connections should also be checked.**

PrRes_Dig_F

The PrRes_Dig_F program steps a bit through each digital register on the PRO board and verifies each written value through read-back circuitry. A Turbo Selftest Assembly is not required.

- **If failures occur in this routine, the problem is almost always in the Programmable Resistor/Ohmmeter board.**
- **Power to the Testhead and connections should also be checked.**

RMux

RMux_f

This program exercises any and all Relay Multiplexer boards present in the system. It checks all the ranges on the boards by placing a voltage supplied by the TDAC on the TSTRLY bus in the Selftest Assembly. The voltage is then measured with the Amplitude Measurement System board through the first channel of each group of sixteen multiplexers and checked to see if it is close to the voltage programmed. It is then corrected for gain and offset. The correction factors are saved and used on the remaining 15 channels in that group. These channels are then checked to see that the voltage read through them is within less than 0.1% on the 10 and bypass ranges, 0.5% on the 1 and 100 ranges, and 5% on the 0.1 volt range.

Initially, the program checks for the presence of a Selftest Assembly, and determines if the Selftest Assembly is a standard or Turbo type. Then the Testhead is searched for any Relay Multiplexer boards, and each board found is identified. The Selftest Assembly is checked for Relay Multiplexer Selftest boards in the proper slots. Then the first channel of each group is tested as described above, and the remaining channels of each group are tested based on the results of the first channel. All Relay Multiplexer boards are tested in order.

- **If failures occur on only one Relay Multiplexer board in a multiple board system, the Relay Multiplexer board is most likely the problem.**
- **If all Relay Multiplexer boards (more than one) in a system fail, check the Amplitude Measurement System and Selftest Assembly for failures.**
- **Power to the Testhead and connections should always be checked.**

RMuxCap_f

The system must be warmed up, and the DMS (or MDMS) must be calibrated prior to running this test. A Turbo Selftest Assembly is required for this test.

This program exercises all RMUX boards in the system to verify that their input filter capacitors are adjusted and functioning properly. The test works by

performing an AC sweep on each RMUX group's 20 and 200 Volt ranges. The test only performed when there is a DMS or MDMS in the system.

- **If failures occur on only one Relay Multiplexer board in a multiple board system, the Relay Multiplexer board is most likely the problem.**
- **If all Relay Multiplexer boards (more than one) in a system fail, check the Anbalog Source Board, DMS/MDMS boards, and Selftest Assembly for failures.**
- **Power to the Testhead and connections should always be checked.**

RMux_dig_f

The RMUX_dig_f program was written to test all the digital circuitry on all Relay Multiplexer boards installed in the system.

When this test is run, the program searches the Testhead for Relay Multiplexer cards. After one or more Relay Multiplexer cards are found, the program writes to and reads from the multiplexers.

- **If a failure is reported, the fault is most likely on the Relay Multiplexer board itself.**
- **Power to the Testhead and connections should always be checked.**

RMux_prot_f

This test exercises the "break-before-make" protection circuitry on the Relay Multiplexer boards.

The program checks for the presence of any Selftest Assembly. The Testhead is then searched for Relay Multiplexer boards. A signal is generated by the TDAC in the Selftest Assembly. This signal is fed through the test relays to a channel in the first group on the first Relay Multiplexer board. The signal is then passed through the Amplitude Measurement System to the Measurement Display Electronics. A sweep is generated by the Measurement Display Electronics board and read by the event counter on the Time Measurement System board.

The signal is switched to another channel of the same group, and the event counter senses a break in the sweep due to the "break-before-make" protection circuitry. The signal is then switched to a channel in the second group and this group is tested in the same manner. This process continues until all groups on all Relay Multiplexer boards in the system are tested.

- **If a failure occurs, the Relay Multiplexer board is most likely faulty.**
- **If all Relay Multiplexer boards in the system fail, check the Amplitude Measurement System, Measurement Display Electronics, and Time Measurement System boards and possibly the Selftest Assembly.**
- **Power to the Testhead and connections should always be checked.**

SMI

smi_dig_f

The smi_dig_f selftest program tests the digital logic on all the SMI boards found in the Testhead. This program will walk a logic '1' through each of the write registers and verify the data input latches by reading back the corresponding data readback register. This test requires a Selftest Assembly to be present because the SMI interlock and disconnect control registers will be held in reset unless +5VDC is applied to the SMI board's +EXT Patchboard pins. This test will also verify the buried registers in the Trigger Matrix PLD in a similar manner.

- **If this test fails, the problem is most likely on the SMI board.**
- **Since the Selftest Assembly is used to control some circuitry during the test, it should also be checked.**
- **Power to the Testhead and connections should always be checked.**

smi_intlk_f

The smi_intlk_f selftest program tests the safety interlock and Patchboard interlock logic on all the SMI boards found in the Testhead. The Patchboard interlock is tested first, then the safety interlocks.

Patchboard Interlock Test

For each SMI supply: The SMI selftest board is setup to close the safety interlock for the selected supply. The board enable relay on the Selftest Assembly is turned on. The selected supply is then turned on and programmed. The actual supply output voltage is measured using the AMS and sig5 measurement path. The board enable relay on the Selftest Assembly is opened and the supply output voltage is measured again. The test should report a failure if the output voltage isn't approximately 0VDC. Finally, the test results are reported using PTA. The test is repeated for each SMI supply configured for the selected SMI board.

Safety Interlock Tests

For each SMI supply: The SMI selftest board is setup to close the safety interlock for the selected supply. The board enable relay on the Selftest Assembly is turned on. The selected supply is then turned on and

programmed. The actual supply output voltage is measured using the AMS and sig5 measurement path. The safety interlock on the Selftest Assembly is opened and the supply output voltage is measured again. The test should report a failure if the output voltage isn't approximately 0VDC. Finally, the test results are reported using PTA. The test is repeated for each SMI supply configured for the selected SMI board.

- **If this test fails, the problem is most likely on the SMI board.**
- **The AMS is used to measure the output voltage and could possibly cause failures.**
- **Since the Selftest Assembly is used to control some circuitry during the test, it should also be checked.**
- **Power to the Testhead and connections should always be checked.**

smu_test_f

This program first tests the SMU sense and guard connections. Next, both the linearity and accuracy of the voltage and current outputs of each of the SourceMeter Units (SMU) are tested. The voltage and current linearity of each SMU is tested first in programmed steps ranging from 5% of full scale to 95% full scale in 5% increments. Then, the voltage and current accuracy of each SMU is tested at 25%, 50%, and 75% of the SMU full-rated output. The following descriptions summarize the requirements for the sense and guard tests:

SMU Sense Test

The SMU sense test verifies that the sense lines of the SMU are connected. This is done by measuring known resistances on the SMU selftest board. During this test, each of the voltage measurement ranging resistors will be connected in turn to the SMU output. The resistance will be measured for each range by the SMU using 4-wire sensing and are compared to the allowable tolerances. The results of each measurement are reported using PTA. The above test may also be done by measuring the current sense resistor on the SMI selftest board.

SMU Guard Test

The SMU guard test verifies that the guard and guard sense lines of the SMU are connected. This is done by measuring the guard test resistor on the SMU selftest board using a 6-wire ohms measurement method. The measured resistance is compared to the allowable tolerances and the results of each measurement are reported using PTA.

SMU Voltage Linearity

The SMU voltage linearity tests are done with no load applied to the supply. The SMU output current compliance limit is programmed to its maximum value allowed. The SMU output voltage is then stepped through its 5% to 95% full scale range in 5% increments. SMU output voltage readings taken by the SMU are compared to the allowable tolerances. The results of each measurement are reported using PTA.

SMU Current Linearity

The SMU current linearity tests are done with a load applied to the supply. The SMU output voltage compliance limit is programmed to its maximum value allowed. The SMU output current is then stepped through its 5% to 95% full scale range in 5% increments. SMU output current readings taken by the SMU are compared to the allowable tolerances. The results of each measurement are reported using PTA.

SMU Voltage Accuracy

The SMU voltage accuracy tests are done with no load applied to the supply. The SMU should be configured to take 4-wire measurements. The SMU output current compliance limit is then programmed to its maximum value allowed. The SMU output voltage is tested at 25%, 50% and 75% of the full-rated output. SMU output voltage readings taken by the SMU are compared to the allowable tolerances. The results of each measurement are reported using PTA.

SMU Current Accuracy

The SMU current accuracy tests are done with a load applied to the supply. The SMU output voltage compliance limit is programmed to its maximum value allowed. The SMU output current is then tested at 25%, 50% and 75% of the full-rated output. SMU output current readings taken by the SMU are compared to the allowable tolerances. The results of each measurement are reported using PTA.

This selftest routine is repeated for each SMI board found in the tester.

- **If this test fails, the problem is most likely on the SMI board.**
- **Power to the Testhead and connections should always be checked.**

vps_test_f

This program tests both the linearity and accuracy of the voltage and current outputs of each of the Voltage Programmable Supplies (VPS). The voltage and current linearity of each VPS is tested first in programmed steps ranging from 5% of full scale to 95% full scale in 5% increments. Next, the voltage and current accuracy of each VPS is tested at 25%, 50%, and 75% of the VPS full-rated output. DA0 is used to control the VPS output voltage setting. ARB0 is used to control the VPS output current setting.

Before the linearity and accuracy tests are performed, the voltage measurement path on the SMI selftest board will be calibrated for each range using TDAC as a source. The following descriptions summarize the requirements for the linearity and accuracy tests:

VPS Voltage Linearity
The VPS voltage linearity tests are done with no load applied to the supply. For the voltage linearity test, ARB0 is set to the voltage needed to program maximum VPS output current. The VPS output voltage is then stepped through its 5% to 95% full scale range in 5% increments using DA0 to program the supply. Voltage readings are taken using the supplies voltage monitor output signal. These measurements are made by the AMS over the sig4 measurement path. These readings are converted to VPS output voltages and are compared to the allowable tolerances. The results of each measurement are reported using PTA.

VPS Current Linearity
The VPS current linearity tests are done with a load applied to the supply. For the current linearity test, DA0 is set to the voltage needed to program maximum VPS output voltage. The VPS voltage is then stepped through its 5% to 95% full scale range in 5% increments using ARB0 to program the supply. Current readings are taken using the supplies current monitor output signal. These measurement are made by the AMS over the sig6 measurement path.

These readings are converted to VPS output currents and are compared to the allowable tolerances. The results of each measurement are reported using PTA.

VPS Voltage Accuracy

The VPS voltage accuracy tests are done with no load applied to the supply. For the voltage accuracy test, ARB0 is set to the voltage needed to program maximum VPS output current. The VPS output voltage is then tested at 25%, 50% and 75% of the full-rated output using DA0 to program the supply. Actual supply output voltage readings are taken using the AMS over the sig5 measurement path. These readings are adjusted by the calibration factors determined at the beginning of this test and are compared to the allowable tolerances. The results of each measurement are reported using PTA.

VPS Current Accuracy

The VPS current accuracy tests are done with a load applied to the supply. For the current accuracy test, DA0 is set to the voltage needed to program maximum VPS output voltage. The VPS output current is then tested at 25%, 50% and 75% of the full-rated output using ARB0 to program the supply. The actual supply output current is determined by taking measuring the voltage drop across the 2 ohm current sense resistor using the AMS over the sig5 measurement path. The actual output current is calculated using these readings and are compared to the allowable tolerances. The results of each measurement are reported using PTA.

This selftest program is repeated for each SMI board found in the tester.

- **If this test fails, the problem is most likely on the SMI board.**
- **The ASB is used to control the supplies during testing and could possibly cause failures.**
- **The AMS is used to measure the output voltage and could also possibly cause failures.**
- **Since the Selftest Assembly is used to control some circuitry during the test, it should also be checked.**
- **Power to the Testhead and connections should always be checked.**

TMS

TMS_dig_f

The TMS_dig_f test was written to test the Time Measurement System event counters ability to preload specified data. This is a strictly digital test and doesn't require a Selftest Assembly or other boards in the Testhead except for a Measurement Display Electronics board.

Initially the program searches the Testhead for the Time Measurement System and Measurement Display Electronics boards. When it finds both, it performs some setup functions on both boards. The program then writes to and reads back from the event counter circuitry. If the data read back doesn't match the data written, a flag is set.

- **If this test fails, the problem is most likely on the Time Measurement System board.**
- **If a Measurement Display Electronics board is not present in the Testhead, an error message will be displayed.**
- **Power to the Testhead and connections should always be checked.**

TMS_event_f

This program tests the event counter on the Time Measurement System board using the Amplitude Measurement System and Measurement Display Electronics board circuitry. No Selftest Assembly is required.

The program initially searches and identifies the Amplitude Measurement System, Measurement Display Electronics, and Time Measurement System boards. Then the trigger circuit on the Amplitude Measurement System board is disabled, and SigA, Sig1, and Sig2 are set to ground. The counter is cleared and the Time Measurement System is initialized and read back. The program completes a series of counter tests using SigA, Sig1, and Sig2 and the event counter on the Time Measurement System board.

- **If errors occur during the test, the Time Measurement System board is most likely faulty.**

- **Since the Amplitude Measurement System and Measurement Display Electronics boards are also exercised, they can also be checked.**

- **Power to the Testhead and connections should always be checked.**

TMS_test_f

This test exercises the Time Measurement System using the internal timebase created by the 100 MHz TCXO to count the same 100 MHz pulses. No Selftest Assembly is required for this test.

The program first identifies the Time Measurement System board in the Testhead. It then generates fifteen frequencies from 1 KHz to 100 MHZ and reads these frequencies back through the counter circuitry.

- **If failures occur, the problem is most likely on the Time Measurement System board itself.**

- **Power to the Testhead and connections should always be checked.**

VPW-J1850

j1850vpw_dig_f

The j1850vpw_dig_f selftest program tests the digital logic on all the VPW-J1850 boards found in the Testhead. This program will walk a logic '1' through each of the write registers and verify the data input latches by reading back the corresponding data readback register. The test will also verify the buried registers in the Trigger Matrix PLD in a similar manner.

- **If errors occur, the VPW-J1850 board is most certainly the problem.**
- **Power to the Testhead and connections should always be checked.**

j1850vpw_ser_f

The j1850vpw_ser_f selftest program functionally tests all of the J1850 channels on each board in the system. This is done by first connecting Channel 0 to Channel 1 together by closing the appropriate switches on the selftest card, then transmitting a one byte message from one channel and receiving it on the other. The test passes if the received message is the same value as the transmitted message. This test is repeated using a walking '1' for a total of eight transfers. The selftest switch for Channel 0 is opened and Channel 1 and Channel 2 are connected and tested. This is repeated until each channel on each VPW-J1850 board is tested.

- **If errors occur, the VPW-J1850 board is most certainly the problem.**
- **Since the Selftest Assembly is used to control some circuitry during the test, it should also be checked.**
- **Power to the Testhead and connections should always be checked.**