1458/1458HP/1458LP/1454
HV Mainframe
Optional 1450-ET Ethernet and 1450-ARC Arcnet Interface

System Test Specification
7-Mar-05

V3.08 - A
50-1U22J
50-1U19J

Table of Contents

Introduction

This document contains the system test specification for the LeCroy 1454 and 1458 High Voltage mainframes.

The version control numbers on the cover page can be further interpreted as the 1450-1 firmware version followed by a dash then a letter indicating the revision level of this specification. Following lines list corresponding EEPROM part number identifications and checksums pertinent to the enclosed documentation. The EEPROM information can be used to further cross-reference to 1450-1 hardware ECO's.

This specification serves two purposes. The first is to document in a systematic fashion all supported features of the 1454 and 1458 mainframes. The second is to provide a step by step test procedure for these features.

Step by step test procedures are given in outline form and designed for daily use by test technicians. Intermixed with the test procedures are introductory paragraphs and/or annotations which discuss in more general terms the features being tested. The annotations (and the 1454/8 User's Guide) are generally more of interest to engineering and marketing personnel since they attempt to document the system features in the absence of a product specification.

Test procedures are separated into various logical categories within these categories are step by step instructions to carry out specific tests. Groups of step by step instructions are generally proceeded by a descriptive heading. The formatting of the descriptive headings indicate the relative importance of a particular test.

Headings which are **underlined and bold** indicate tests which should be performed on every mainframe unit. Headings which are **bold** indicate tests which should be performed on at least one mainframe in a manufacturing lot. Other tests need only be performed when a major mainframe firmware release has occurred.

The last major section of this specification documents the operation and significant contents of the remote interface test program, hvtest.exe.

Test Equipment Requirement

Listed below is the test equipment required to carry out all of the tests listed in later sections.

The Mainframe Initiation and Remote Interface Scripts sections require most of external test equipment. Tests in other sections can generally be carried out with the addition of a VT100 terminal and a null modem cable (especially needed for the 1458) along with one or more 1461N HV modules.

Hardware

1. HV mainframe with more than one 1461N module.
2. (2) SHV T's for removing modules.
3. 386 PC with at least 4 MB of hard drive space.
   A. Running DOS 6.0 (or greater)
   B. Available serial port (9 pin)
   C. Ethernet interface card, P/N 1450-ET
   D. Low speed TTL I/O card BNC I/O ports.
4. Cables
   A. 1 - 12' Null Modem Serial Cable (9-pin)
   B. 1 - 1000' Mile 90 Ohm RG-62 Cable
   C. 1 -12' 90 Ohm RG-62 Cables
   D. 2 - BNC T's
   E. 3 - 6' BNC cables (Status, Macro, AC Power Control)
   F. 1 - 50 Ohm BNC Terminator
   G. 2 - SHV T's (1450 module extraction tools)
   H. 2 - 90 Ohm BNC Terminators
5. 13 - 1458 Power Supply Load Cards P/N: 1589-58 (For 1458xx Only)
6. 3 - 1454 Power Supply Load Cards P/N: 1589-54 (For 1454 Only)
7. Extra PC keyboard to plug into HV Mainframe
8. VT100 or VT200 terminal or additional PC running VT100 terminal emulator.
9. LeCroy scope with histograming capability. (For serial baud checks, etc. ?)
10. DVM
11. TTL controlled AC power switch. (Remote Interface Scripts).
12. Additional VGA monitor and interface card (1458 - Mainframe Initiation).
13. 1 MOhm HV load able to tolerate 2KV.
14. For testing 1450-ET Ethernet Interface option:
    A. Mitron Ethernet ISA card LX-2000JL (LeCroy 1450-ET)
    B. 1 - 6' BNC 50 Ohm Cable
    C. 2 - BNC T's
    D. 2 - 50 Ohm Terminators

Software

The test programs, hvtest.exe and bsdtest.exe, issues HV mainframe commands via RS-232 or ARCNET to 1454/8 HV mainframe and checks responses based on compiled script functions. This program also supports user input to issue individual commands or a file containing a list of commands.

Test Configurations

Typically, test procedure outlines in later sections begin with a "setup" step of the form "Configure for
Setup-name", which refers to setup procedures documented in this section. Each major (numeral) heading (bold) corresponds to a particular setup.

1. **All Tests.**
   A. Check 1450-1 Jumper settings and configuration switch setup.
      1. Configuration switch setting for Backplane baud rate of 38.4K
      2. Interlock input configured for "fail -safe" mode. (Currently Hardwired in this manner)
   B. Terminate Interlock with 50 Ohm terminator.
2. **Mainframe Initiation.**
   A. Configure for **All Tests.**
   B. At least one HV module installed.
   C. 1454 - Remove top cover.
   D. 1458 - Remove front cover.
   E. Connect external Keyboard to HV Mainframe.
   F. 1454 - Remove front panel display interface (1450-2) from ISA bus
   G. Install VGA interface card and connect VGA terminal.
3. **Manual Testing.**
   A. Configure for **All Tests.**
   B. Mainframe with more than one 1461N module.
   C. Power up HV Unit (wait until Ready).
      1. Remote/Network LED's stop flashing.
      2. Issued Ready beep code (See 1454 manual):
         a) 1 beeps upon Power up.
         b) then more than 10 sec later 1 beep - Normal , 2 beeps - Warning, or 4 beeps (2 then 2) - Error.
   D. Front Panel Switch in Local.
   E. Panic Off should not be active.
   F. Set all module demand voltages for all channels to -1000V and enable them.
   G. Leave HV Off.
4. **VT100 Testing.**
   A. Configure for **Manual Testing**.
   B. Connect VT100/200 terminal via null modem cable to 1454/8 serial port.

C. Check communication setup of terminal agrees with 1454/8 setup (Factory default baud rate: 9600).
   1. 1458 - Check external baud rate jumper settings (see figure in Mainframe Initiation Section).
   2. 1458 - configure terminal for 8 data, 1 stop, and parity none.
   3. 1454 - Use *SYSTEM* menu and select serial setup menu to observe settings.

D. Set 1454/8 front panel key switch to REMOTE.

E. Establish HV RS-232 command session.
   1. Hit one or two carriage returns, <CR>,to get prompt "0\Enter 1450 to begin>".
   2. In response to this prompt type "1450" (before the time-out).

F. Observe prompt "1\EDIT\1450>".

G. Notice: Remote LED flickers during serial communication.

H. (Initiate VT100 full screen mode as indicated by test, by entering "VT100" to prompt.)

5. HVON ACFail, Restore HVON Testing.

A. 1458 - Configure for **VT100 Testing.**

B. 1454 - Configure for **All Tests**.

C. Power up Mainframe, wait for ready.

D. 1458 - Start VT100 mode.
   1. Initiate external serial command session.
   2. Set front panel switch in remote.
   3. Start VT100 display via "VT100" command.

E. Via System Menu select System Default and set "HVON ACFail, restore HVON Yes".

F. Save these settings by selecting "Save" (PC Key <Alt-S>, VT100 key < S>).

G. Note save complete message.

H. Turn HVON.

6. **Command Script Testing.**

A. Configure for **All Tests.**

B. Mainframe with more than one 1461N module.

C. Power up HV Unit (wait until Ready).
   1. Remote/Network LED's stop flashing.
   2. Issued Ready beep code (See 1454 manual):
   3. 1 beeps upon Power up.
   4. then more than 10 sec later 1 beep - Normal , 2 beeps - Warning, or 4 beeps (2 then 2) - Error.

D. Front Panel Switch in Remote and Panic Off should not be active.

E. Connect Com port via Null modem 9-Pin serial cable of test PC to HV mainframe.

F. Connect ARCNET port of test PC to HV mainframe via 90 Ohm Cable with 90 Ohm Terminators on both ends.

G. Connect PC TTL I/O port (2 and 1) via BNC cables to mainframes STATUS output and MACRO inputs, respectively.

H. Connect PC TTL I/O port (0) to TTL controlled AC power switch and run mainframe power through this switch.

7. **Ethernet Port Testing - 1450-ET.**
   A. Configure for **Command Script Testing.**
   B. Connect Ethernet port of test PC to HV mainframe via 50 Ohm cable with 50 Ohm Terminators on both ends.

Mainframe Initiation

This includes procedures to initialize the 1450-1 as well as those tests which require the 1454 top cover to be removed, the 1458 front panel to be removed, a VGA monitor (1458), or a keyboard.

Procedures

Configure for **Mainframe Initiation**.

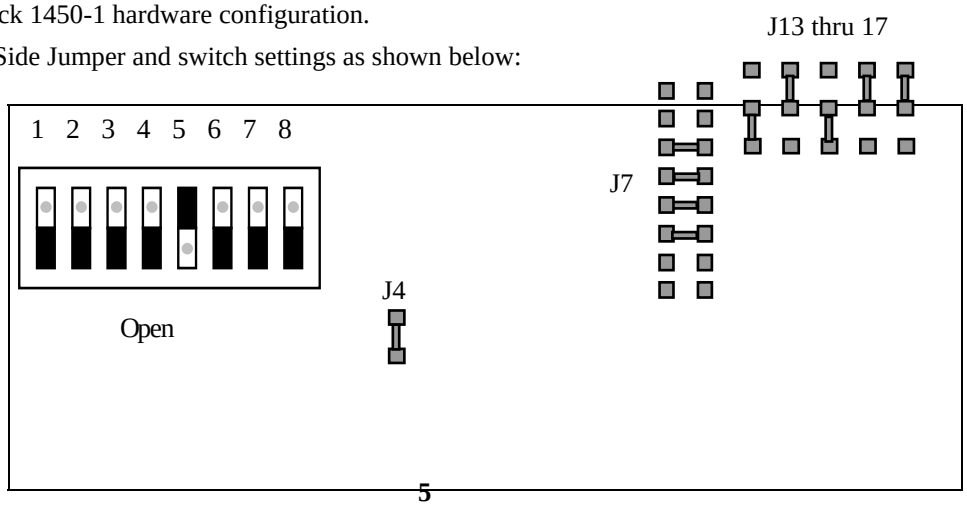
1. Check for correct CMOS configuration

A. Check that proper BIOS chip is installed in motherboard.
   1. Cycle Mainframe power and observe boot sequence.
   2. Observe VGA display ROM BIOS message contains the line "LeCroy 1454/8 486/SLC" (for mainframes with 486 motherboard) above the memory test/scan line and that the amount of memory detected is "3968 KB" (for 386 motherboards this is 4096KB).
   3. The message "Extended Memory mismatch…" is normal for 486SLC boot up.

B. Check that the unit recovers from a loss of CMOS memory.
   1. Turn mainframe power off.
   2. Clear mainframe CMOS memory via JP4 by momentarily connecting Pins 1&2 then putting the jumper back to 2&3. (For 386 motherboards used J2 Jumper and connect 3&4 momentarily).
   3. Turn mainframe power on and observe message complaining about wrong date/time.
   4. Enter CTRL-Break via the keyboard right Immediately after "High Voltage Mainframe" appears on the screen to get a "D:\>" prompt.
   5. To D:\> prompt enter "date" and observe date is totally wrong (indicates CMOS memory loss).
   6. Enter proper date and time.
   7. Cycle mainframe power and observe boot sequence. (There should be no date/time warning messages).

2. Check HW configuration

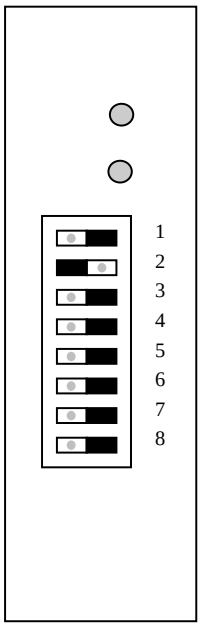
A. Check 1450-1 hardware configuration.
   1. Side Jumper and switch settings as shown below:



2. Connections to motherboard speaker and reset pin headers.
3. Interlock input connected to 50 Ohm terminator.

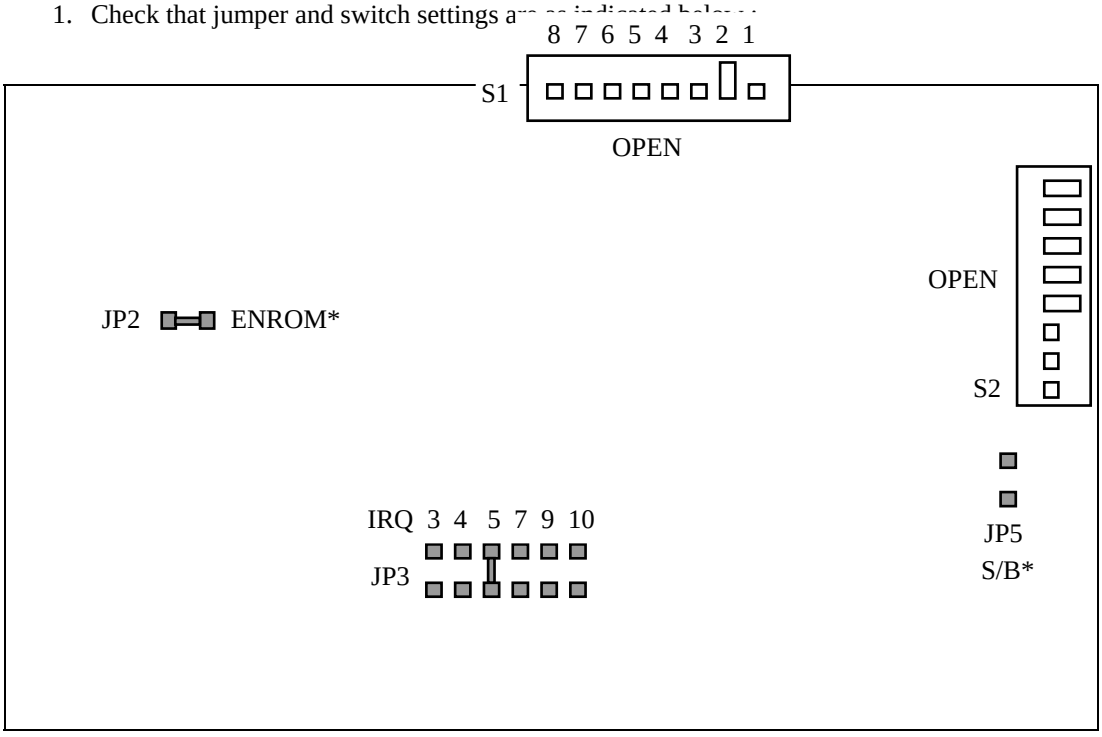
B. Assure that 1450-8 card properly installed and daisy chained on 40 pin ribbon cable also connected to the backplane and 1450-1.

C. 1458 - Check 1450-8 switch block set for 9600 (ISA card with two green LEDs) as shown below:



A. Check ARCNET card configuration.

1. Check that jumper and switch settings are as indicated below:

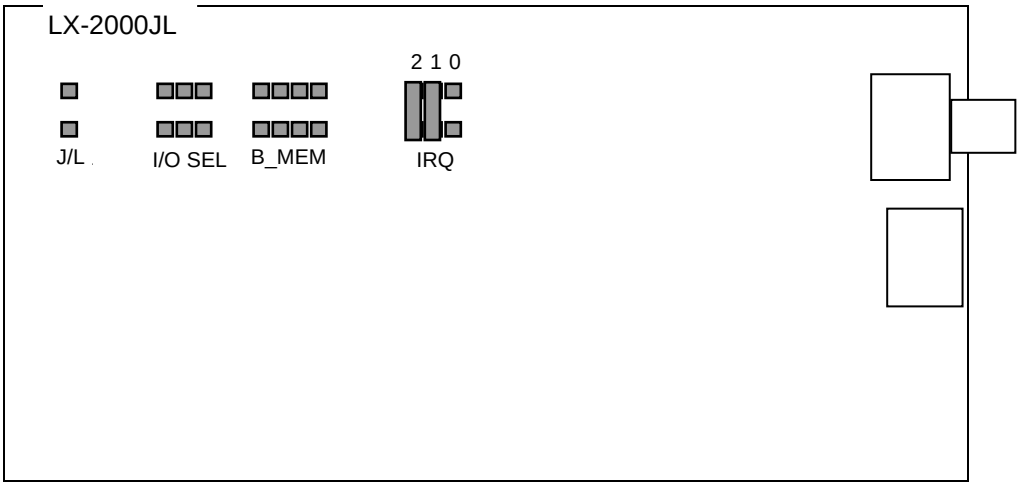


2. Front panel BNC should be connected to BNC Tee and 90 Ohm terminator.

2. Installing Ethernet Option

A. Configure Mitron Ethernet Card LX-2000JL

1. J/L - Open → "Not Jumperless".
2. I/O SEL - 0, 1, and 2 Open → Base Address = 0x300.
3. B_MEM - 0, 1, 2, and 3 Open → Boot Memory Address Disabled
4. IRQ - 0 Open, 1 Closed, and 2 Closed → Interrupt Level = 12.



A. Verify Card recognizes 10BaseT (RJ-45) connection (if active 10BaseT network available)

1. Plug 10BaseT into active 10BaseT network.
2. Turn On power observe the green LED is on.
3. Green LED goes off when network is disconnected from card.
4. Note: LED does not indicate proper jumper configuration. Also LED will not come on when connected to 10Base2 (BNC) network.

2. HV mainframe low level Tests

A. Power on unit, Immediately after "High Voltage Mainframe " is displayed type Ctrl-Break.

B. Observe DOS prompt "D:\>".

C. To the DOS prompt enter "lhv yd" and observe test results.

1. failures generally indicate low-level 1450-1 or 1450-8 failures.
2. Typically the low level EEPROM test will fail on a new HV mainframe (1450-1).

3. To fix EEPROM failure, zap the Mainframe EEPROM

A. WARNING: This procedure *clears* the mainframe serial number and other mainframe specific information in the system information display.

B. Turn on Mainframe.

C. *Immediately* after "High Voltage Mainframe" is displayed, type <Ctrl-Break>.

D. DOS prompt "D:\>".

E. To the DOS prompt enter "lhv z"

F. Observe message that EEPROM cleared successfully.

4. <u>Check operation of 24V Bad.</u>

A. Note that older 1454's do not have this feature.

B. Unplug AC power from back of unit.

C. Ground 24V_GOOD signal.

1. 1454 - Temporarily ground pin 2 on 1454 backplane of 10 pin header on solder side.
2. 1458 - Disconnect 10 pin cable from AC controller card (1458-5). Insert a small wire into the female pin 2 of this connector and ground the other end of this wire.
3. Check for operation of 24V bad signal.

D. Plug AC power back into unit and turn on unit.

E. After power up complete, observe "24V Bad ...." message is displayed.

F. Unplug AC power from back of unit.

G. Unground 24V_GOOD signal.

1. 1454 - Remove temporary grounding of pin 2 on 1454 backplane.
2. 1458 - Remove small wire from connector and reconnect to AC controller card.

H. Plug AC power back into unit and turn on unit.

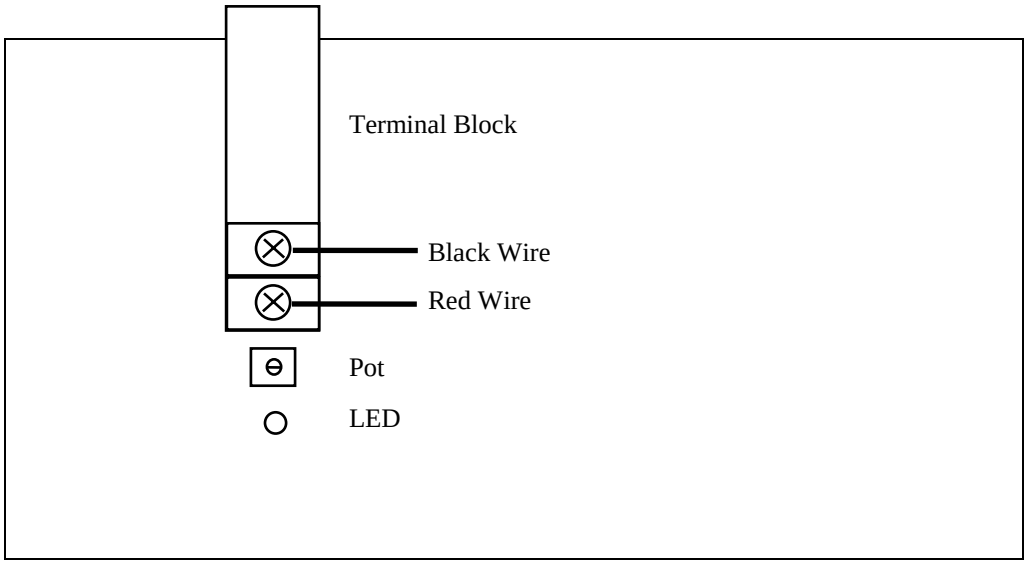
I. After power up complete, confirm that "24V Bad ...." message is not displayed.

5. <u>1458 - Set Housekeeping 21V</u>

A. Locate Housekeeping supply -

1. The Housekeeping supply is located just to the right of the 1458 -5 AC controller card discussed above.

1. See diagram below.



B. Use DVM to monitor voltage of screws on housekeeping supply terminal block that hold the red and black wires. The red wire is plus.

C. Using pot on supply, adjust voltage indicated on the DVM to +21V

2. Confirm Panic Switch and Interlock Hardware work.

A. Turn On Power, Hit Panic Off and observe red Panic Off LED's.

B. Wait for mainframe to be ready (when the remote/network LED's stop flashing).

C. Put the front panel switch in LOCAL,.

D. Reset Panic Off (via 1454 - front panel menu selection or HVOFF).

E. Remove Interlock termination.

F. Observe red "Interlock Error" LED's and Panic Off LED's.

G. Replace Interlock termination.

H. Observe Panic Off LED's on and Interlock LED's are off.

I. Reset Panic Off.

3. Affix Serial Number sticker to Mainframe and note number.

4. Enter Mainframe System Information.

A. Power up Mainframe, wait until ready.

B. Set EEPROM write configuration switch (2nd switch from the rear end of the 1450-1 ISA card switchblock) to "ON" or "OPEN". (See figure under Item 2.).

C. Type "s" to activate SYSTEM menu, and select "System Information".

D. Use keyboard to enter Mainframe serial number, revision level , ECO level, tested by, and test date information . (For example on 9/27/95 the revision level was "B" and the ECO level was "1103".)

E. Exit System Information menu via <ESC>.

F. Set EEPROM write configuration switch back to "OFF" or "CLOSED" on the 1450-1.

5. Confirm Mainframe System Information .

A. Power down Mainframe via rear panel rocker arm switch .

B. Power up Mainframe

1. Make sure Mainframe does not "double boot". This condition is indicated by twice the initial motherboard beep sequence. In particular, 1 or 4 beeps (2 beeps, pause, beep, pause, beep) then a pause then 1 or 4 beeps again (before the 1454 display lights up and the REMOTE/NETWORK LED's start flashing).
2. In startup display
   a) Observe line "Turbo Update Installed" indicating 1450-8 installed.
   b) Observe proper mainframe serial number.
   c) Observe FW Version is most recent release.
   d) 1458 - Observe External serial baud is 9600 (controlled by 1450-8 switches).
3. After system is ready, type "s" to activate SYSTEM menu and select System Information.
   a) Verify System Information is correct.
   b) Check date and time is correct on bottom of Power up menu by selecting 0)

6. <u>Error Beep Sequence and watchdog reset checks (requires keyboard).</u>

A. **<u>Control Program Crash (this is only test needed for every mainframe)</u>**

1. Turn power on wait for ready.
2. Cause system crash via keyboard <Ctrl-Break>.
3. Observe Mainframe reset and reboot.
4. After reboot, observe beep sequence after network/remote flashing ends, 4 beeps (2 then 2).
5. 1454 - Notice Power up status “hung error” message.
6. 1458 - Notice same message on VGA screen, but no pause for user input.

B. Control Program Multi-tasking Hung

1. Set main spreadsheet display to observe heart beat character.
2. Cause crash via keyboard <Alt-k>.
3. Observe heart beat stop,.
4. Observe delay of 20-40 seconds then mainframe reset and reboot.
5. After reboot, observe beep sequence, after network/remote flashing ends, 4 beeps (2 then 2).
6. Notice Power up status “hung error” message.

C. Control Program exit on error with HVON.

1. Set display to observe main display sheet and turn HV ON.
2. Notice *module* front panel red LEDs are ON.
3. Cause internal exit on error <Alt-j>.
4. Notice *module* front panel red LEDs blinking then go off (HV ramp down).
5. Observe box in center of screen with error message.
6. Observe 4 repeating beeps, blinking “HV ERROR” and flickering front panel “HVON”.
7. Observe beeping and LED blinking continues for 15-20 seconds, then mainframe reboots.
8. After reboot, observe beep sequence, after network/remote flashing ends, 4 beeps (2 then 2).
9. Notice Power up status “parameter out of range” message.

D. Control Program exit on error with HV OFF

1. Set display to observe large 3-line display (via system menu) and leave HV off.
2. Cause internal exit on error, this time pull out an HV Module while the mainframe is running.
3. Observe same effects as in previous test (C.) except no HVON LED flashing with error message box now in a graphics font in 3-line display.
4. Observe error message indicating communication lost with module.

7. Mainframe Hardware problem Beep Sequence (requires front cover off).

A. Unplug 1450-8 from 40 pin ribbon cable daisy chain.

B. Turn power on.

C. Observe beep sequence, 1 beep then repeated 3 beeps with a 5 second delay.

8. **Restore HVON after ACFail Test 5 - Disabled by system error (requires keyboard).**
   A. Configure for **HVON ACFail, Restore HVON Testing** (See Page 4)..
   B. Turn on HV via HVON button.
   C. Cause system crash via keyboard <Ctrl-Break>.
   D. Wait for system reboot.
   E. Observe HV remains off.
   F. 1454 - Observe message that system default changed.
   G. Check System Default menu , "Restore HVON after ACFail NO".
   H. Leave mainframe configured HVON ACFail, Restore HVON No".
9. **Check that the ARCNET card jumper JP-5 has been removed.**
10. **Check that backplane connectors have all nuts and bolts removed or that connector nuts are on solder side of backplane.**
11. **1458 - Tie wrap internal cables and wires to prevent fan interference.**
12. **1458 - Check clearance between front panel and power support tray.**
    A. Adjust support tray stops as needed to ease pressure on front card when front panel grill installed.
    B. Note that stops must also prevent rear of 24V power supplies from hit fans.
13. **Tie wrap 1450-1 to motherboard.**
14. **RTV Motherboard power connector, speaker and reset wires.**
15. **Check that motherboard ground is connected.**
16. **Mainframe Initiation close-up**
    A. Remove video card from Motherboard and PC keyboard.
    B. 1454 - re-install 1450-2.
    C. 1458 - install front panel grill.
    D. 1454 - install top cover.
    E. All other testing is done via front panel or serial/VT100 interface.

Annotations

Especially for a 1458, it is important that the mainframe issues only a single beep while the CPU is booting additional beeps indicate the CMOS setup is incorrect or some more severe motherboard problem.

Low level diagnostics are included as part of the mainframe control software which mostly check the previously tested 1450-1 card . Since these tests might be used in the field, it seems prudent to assure they pass as installed in a particular mainframe.

The necessity to "zap" the EEPROM may arise if a new version of the control software is installed on a 1450-1 that successfully ran a older version of the control software using presumably fewer EEPROM storage locations.

The Interlock input and it LED's and the Panic off relay switch and LED's are hardware driven objects. Basic interlock and panic off checks are done here, while features controlled by software are tested later.

Both user save sets and operation log entries are annotated with the mainframe date and time, these should be set to the local time.

The mainframe preserves in EEPROM storage the mainframe serial number and other test information These can be viewed in the “System Information” menu.

Some mainframes can have an incorrectly setup hardware reset or a CMOS configuration, which causes them to “double boot” before the (1454 display lights up or the network/remote LED’s start flashing). This should be fixed since it slows down the power up of the unit.

Additional tests in this section are those requiring a keyboard and/or a VGA display (1458). There are distinctive Power up beep sequences for different error conditions. By crashing the program via the keyboard, we check not only these beep sequences but also the CPU watchdog which should reboot the system when the control program crashes (Checks motherboard/1450-1 reset connection.)

Further reset checks test various ways that a system reset may occur along with the warning messages and/or LED flashing that accompany them. These tests need only be done upon a new FW release.

For safety in the event of a program crash, the “Restore HVON after ACFail” feature is disabled until explicitly re-enabled by the user.

If the JP-5 jumper is not removed from the SMC ARCNET card the BNC port is not properly terminated. This lack of proper termination will probably not show up in later tests since the BNC cable connections between the PC and HV mainframe ARCNET ports are very short.

Local Interface Tests

Overview

The following test procedures exercise the local interface of the LeCroy 1454 and 1458 (HV) mainframes. These tests unfortunately must be done by hand to completely exercise all of the local interface features of the mainframes.

Safety Features

The following procedures exercise all features of Panic Off either caused by hitting the panic button or by setting the Interlock input. The remote/local switch is tested to assure remote or local change of values or HVOFF to HVON transition locked out. The "Restore HVON after ACFail" is tested including safety issues associated with this feature.

Panic Off

Procedures

Configure for **Manual Testing**

1. **Panic Off Test 1 - Panic Off prevents HVON and other changes**
   A. Set front panel switch to LOCAL.
   B. Hit *PANIC OFF*, Observe Panic Off LED's.
   C. 1454 - Observe automatic menu panic reset menu, select reset Panic Off later.
   D. Hit HVON,
   E. 1454 - Observe error message and HV remains off.
   F. Use *HVOFF* to clear Panic Off condition, Observe Panic Off LED's.
2. **Panic Off Test 2 - Panic Off cannot be reset in REMOTE, only in LOCAL.**
   A. Set front panel switch to REMOTE.
   B. Hit *PANIC OFF*, Observe Panic Off LED's.
   C. Hit HVOFF and observe Panic is not reset.
   D. 1454 - Observe error message.
   E. Set front panel switch to LOCAL.
   F. Hit HVOFF and observe Panic Off is reset.
3. **Panic Off Test 3 - Panic Off turns off HV**
   A. Set front panel switch to LOCAL.
   B. Hit *HVON*, wait for channel HV to ramp up and HVON light steady ON.
   C. Observe HV module LED's (on module front panel) first blink during ramp then steady ON.
   D. Hit *PANIC OFF*, Observe HVON flash then go off. Observe HV module LED's (on module front panel) go off. (Note: The modules will flash longer if a target Voltage is set with a slow ramp rate.)
   E. 1454 - Select to reset Panic Off now.
   F. 1458 - Reset Panic Off with *HVOFF*.

4. **Panic Off Test 4 - Panic Off via Interlock**
   A. Set front panel switch to LOCAL.
   B. Panic HV mainframe by removing 50 termination from Interlock Input.
   C. Observe Interlock LED on.
   D. 1458 - Attempt to reset Panic Off with *HVOFF*
   E. 1454 - Attempt to reset by selecting reset Panic Off now.
   F. Observe Panic Off does NOT reset.
   G. 1454 - Observe message that Interlock is active.
   H. Replace Interlock input terminator, observe Interlock LED go off.
   I. 1454 - Select reset panic now in menu.
   J. 1458 - Use *HVOFF* to reset Panic Off.
5. **Panic Off Test 5 - Panic Off remembered after Power Down**
   A. Set front panel switch to LOCAL, then Hit Panic Off button.
   B. Turn off HV mainframe power via rear panel rocker switch.
   C. Turn on power, observe mainframe still has Panic Off active.
   D. Observe that after Remote/Network LED's stop flashing that the mainframe issues two beeps.
   E. 1454 - Select menu option to reset Panic Off now.
   F. 1458 - Reset Panic Off condition via HVOFF button.

Annotations

After the mainframe panic button has been hit, the panic off is said to be active or panic'ed. When Panic Off is active, not only is the user inhibited from turning on the HV but he is also prevented from making other setting changes.

The panic off feature is considered to be a local feature which means it can only be initiated locally and reset locally. As a result the following safety feature is supported. The user can switch an already panic'ed unit to remote then remove the key to prevent someone (without the key) from resetting the panic off condition.

It is important to observe that not only does the mainframe indicate that a HVON state is switched to OFF when PANIC off is active but that the modules also turn off (their LED's go out).

The panic off active condition that is caused by a Interlock input signal cannot be reset until the Interlock input has been restored. Currently all mainframes require this input to be "closed" or shorted (through a terminator) to prevent a panic off. Future mainframes may have this input configurable as either open or closed sets the panic off active. Even after the initial interlock signal is restored, the panic must be explicitly reset by the user.

A panic off active condition is remembered between power downs. (The panic switch is wired to a mechanical relay.)

Remote/Local front panel key switch

Procedures

Configure for **VT100 testing** (both 1454 and 1458).

Type “q” to get back to command line mode.

1. **Key Switch Remote Test**
   A. Set front panel key switch to Remote.
   B. Hit <CR> and observe prompt to be “1\EDIT\1450>“.
   C. Attempt to turn on HV via front panel *HVON* button, HV should remain Off.
   D. 1454 - Observe error message.
   E. Enter command “HVON”, observe HV On condition.
   F. Use front panel *HVOFF* button to turn off HV (safety feature.)
   G. Hit PANIC OFF, observe Panic Off condition.
   H. Hit <CR> and observe prompt to be “4\PANICOFF\View\1450>“.
   I. Attempt to turn on HV via command “HVON”, observe error message and HV Off condition.
   J. Attempt reset Panic Off condition via *HVOFF* button.
   K. 1454 - Observe error message.
   L. Set front panel key switch to Local, and reset Panic Off condition via *HVOFF* button.
   M. Set front panel key switch to Remote and enter “hi” then <CR>, then observe response then the prompt to be “1\EDIT\1450>“.
2. **Key Switch Local Test**
   A. Set front panel key switch to Local.
   B. Hit <CR> and observe prompt to be “3\LOCAL\VIEW\1450>“.
   C. Attempt to turn on HV via command “HVON”, observe error message and HV Off condition.
   D. Use front panel HVON button to turn on HV.
   E. Enter command “HVOFF”, observe HV Off condition. (safety feature).

Annotations

The front panel key switch serves not only to turn on or off the mainframe power but also define a operating mode either local or remote. When the key switch is in remote only devices connected to the RS-232 or ARCNET can change settings or turn on the HV. When the key switch is in local devices connected to the RS-232 or ARCNET ports cannot change settings or turn on the HV.

Turning the HV off via remote command or the local front panel HVOFF push button is always allowed independent of the front key switch position. However, using the local front panel HVOFF button to reset a Panic Off condition is only permitted when the key switch is in local. (This allows a user to lock in a Panic condition switching the key to remote and removing the key.)

HVON ACFail, Restore HVON

Procedures

Configure for **HVON ACFail, Restore HVON Testing**

Set at least 1 channel's target voltage to some value (100V).

1. Restore HVON after ACFail, Test 1 - Exercise the option. (Exercised by hvtest.exe)
   A. Turn on HV.
   B. Turn mainframe power off via rear panel rocker switch.
   C. Turn mainframe power on via rear panel rocker switch.
   D. After mainframe is ready, observe the following:
      1. 1454 - Warning message on display.
      2. 10 beep countdown.
      3. HVON LED's flash during countdown.
      4. HVON LED's flicker (indicating HV ramping)
      5. HVON LED's on (ramping complete).
2. **Restore HVON after ACFail, Test 2 - Inhibited with front panel Standby.**
   A. Confirm that "HVON ACFAIL, Restore HVON Yes" is configured in system defaults.
   B. Turn on HV.
   C. Turn mainframe power Off via front panel switch to standby.
   D. Turn mainframe power On via front panel switch.
   E. After mainframe is ready, observe HV remains Off.
   F. Observe "HVON ACFAIL, Restore HVON Yes" in system defaults.
3. **Restore HVON after ACFail, Test 3 - Abort HVON countdown.**
   A. Confirm that "HVON ACFAIL, Restore HVON Yes" is configured in system defaults.
   B. Turn on HV.
   C. Turn mainframe power off via rear panel rocker switch.
   D. Turn mainframe power on via rear panel rocker switch.
      1. After mainframe is ready,
      2. 1454 - observe warning message on display.
   E. 1454 - Hit any key to abort HVON countdown.
   F. 1458 - Hit HVOFF to abort HVON countdown.
   G. Check that System Default is still "HVON ACFAIL, Restore HVON Yes".

4. **Restore HVON after ACFAIL, Test 4 - Disabled upon new module configuration.**
   A. Turn on HV.
   B. Turn mainframe power off via rear panel rocker switch.
   C. Remove a single module.
   D. Turn mainframe power On via rear panel rocker switch.
   E. Observe HV remains off.
   F. 1454 - Observe message that system default changed.
   G. Check System Default menu is set "HVON ACFAIL, Restore HVON No".
   H. Reset this option "HVON ACFAIL, Restore HVON Yes" for next test.
   I. Turn mainframe power off via rear panel rocker switch.
   J. Insert the module that was removed.
5. **Restore HVON after ACFAIL, Test 5 - Disabled by Panic Off.**
   A. Turn on HV.
   B. Hit *PANIC OFF* button.
   C. Turn off AC power via rear panel rocker switch, turn on AC power.
   D. 1454 - after mainframe is ready, select to reset panic off later.
   E. 1454 - observe message that system default changed.
   F. Check System Default menu is set "HVON ACFAIL, Restore HVON No".
   G. Switch to local and reset panic off via HVOFF.
6. **Leave mainframe configured for "HVON ACFAIL, Restore HVON No".**

Annotations

This feature is potentially dangerous. So there are number protections to make its use as "safe" as possible. The HV mainframe knows the difference between a front panel (key switch) power down and a rear panel (rocker switch) power down or AC failure. When "HVON ACFAIL, Restore HVON" is enabled (yes), the mainframe only turns on HV if there was an AC power failure (or rear panel rocker switch power down) and the HV was ON at the time power was removed.

Notice that "HVON ACFAIL, Restore HVON" is automatically disabled under a number of conditions and that the user is explicitly required to re-enable this feature. The aforementioned conditions include: after a new module configuration is detected, when the unit is power up with the Panic Off active, and after an unexpected mainframe system crash (tested in Mainframe Initiation).

Powerup Beeps

Procedures

Setup for **Manual Testing**.

1458 - Setup for **VT100 Testing**.

1. **Normal Beep sequence**
   A. Turn power on and wait for ready
   B. 1458 - Initiate VT100 serial session
   C. Change a single channel's voltage.
   D. Turn power off, then on.
   E. Observe beep sequence. 1(or 4) beep(s) then, after network/remote flashing ends, 1 beep.
2. **Warning Beep Sequence - new module configuration**
   A. Turn power off and remove module from backplane.
   B. Turn power on
   C. Observe beep sequence, 1 (or 4) beep(s) then, after network/remote flashing ends, 2 beeps.
   D. 1454 - Observe message that new module configuration found.
3. **Warning Beep Sequence - no module configuration found.**
   A. Delete power up save files.
      1. Push SAVE, then SELECT "Clear Powerup Save Set".
      2. Observe return to Channel vs. value display.
   B. Turn power off and then on.
   C. Observe beep sequence, 1 (or 4) beep(s) then, after network/remote flashing ends, 2 beeps.
   D. 1454 - Observe message that no powerup save files found.
4. **No Modules Installed Beep Sequence**
   A. Turn power off and remove all modules.
   B. Turn power on.
   C. Observe beep sequence, 1 (or 4) beep(s) then repeated 2 beeps with a 5 second delay.
5. Hardware error Beep Sequence
   A. See Mainframe Initiation sequence.
   B. Observe beep sequence, 1 (or 4) beep(s) then repeated 3 beeps with a 5 second delay.

Annotations

In particular for a display-less 1458, the power up beep sequences provide powerup status information that can indicate possible hardware failures, the loss of previous module settings when new module configurations are detected, the fact that no modules are installed, or the mainframe reset itself after an unexpected system crash (tested in Mainframe Initiation).

Manual Key Check and Value Entry

Procedures

Setup for **Manual Testing**

1458 - Setup for **VT100 Testing**

Turn power on.

1458 - Put front panel key switch in REMOTE.

1458 - Begin VT100 session.

1454 - Put front panel key switch in LOCAL.

The 1454/VT100 main display should be Channel vs. Value.

1. Enter a value.
   A. Move cursor over a channel's target voltage.
   B. Push *1 2 3 4* then *ENTER.*
   C. Observe value change.
2. Escape from value entry.
   A. Move cursor over a channel's target voltage.
   B. Push *1*, then *ESC* (Use *e* for VT100).
   C. Observe, channel's target voltage not changed.
3. **Enter a range of values.**
   A. Move cursor over a channel's target voltage
   B. Push *SELECT*, then cursor down to highlight 3 or 4 channels.
   C. Type *1 5 6 7* then *ENTER*.
   D. Observe range of values change.
4. Enter a module of values.
   A. Move cursor over the first channel of a module's target voltage
   B. Push *SELECT*, then push *NEXT PAGE*.
   C. Type *1890* then *ENTER*.
   D. Observe values all channels of module change.
5. Delta a value (increase).
   A. Move cursor over a channel's target voltage.
   B. Push *DELTA*, then push *8 9* then *ENTER*.
   C. Observe value change by delta (increase).
6. **Delta a module of values (decrease)**
   A. Move cursor over the first channel of a module's target voltage.
   B. Push *DELTA*, then push *NEXT PAGE*, then *+/-*, then *100, ENTER.*
   C. Observe module of values change by delta (decrease).
7. Increment a value.
   A. Move cursor over a channels target voltage.
   B. Push *INCREMENT* four times, then *ENTER*.
   C. Observe value increase by 4 units in place value of increment.

8. Move place value of increment then increment.
   A. Move cursor over a channels target voltage.
   B. Push *SELECT*, then move cursor in edit value region to 100's place.
   C. Push *INCREMENT* 3 times, then *ENTER*.
   D. Observe value increase by 300.
9. Decrement a value.
   A. Move cursor over a channels target voltage.
   B. Push *DECREMENT* 2 times, then *ENTER*.
   C. Observe value decrease by 2 units.
10. Save current values
   A. Push *SAVE*, then push *1*.
   B. Observe Save successful message.
   C. Hit any key to continue.
11. Restore values
   A. Change some value from previous save operation.
   B. Push *RECALL*, then push *1*.
   C. Observe Restore be patient message.
   D. Observe Restore successful message.
12. Enable a channel
   A. Move cursor over a (disabled) channel's measured current.
   B. Push ENABLE.
13. Disable a channel
   A. Move cursor over a (enabled) channel's Ramp Up rate.
   B. Push DISABLE.
14. Turn on HV
   A. Enable a few channels and set some target voltages.
   B. Starting with HVOFF, Push HVON.
   C. Observe flashing HVON LED's.
   D. Observe any enabled channels ramp up to target voltage.
15. Turn off HV
   A. Starting with HVON, Push HVOFF.
   B. Observe flashing HVON LED's.
   C. Observe any enable channel ramp down to 0.
16. Next and Previous display sheet.
   A. (Starting from main channel vs. value display), push *NEXT*.
   B. Observe Module Info Display.
   C. Push *PREVIOUS*.
   D. Observe channel vs. value display.
   E. Push *PREVIOUS* repeatedly to observe all display sheets.
   F. Continue pushing PREVIOUS until channel vs. value display returns.
17. 1454 - Hardcopy , Group Display Dn disabled.

A. Push *HARDCOPY*, *GROUP*, or *DISPLAY DN*.

B. Observe feature not implemented message.

C. Do steps A and B for each key.

18. 1454 - System Menu

A. Push *SYSTEM*, observe system menu.

B. Observe system LED light.

C. System options tested later.

19. 1454 - Chan button.

A. Push *SYSTEM*, observe system menu.

B. Push *CHAN*, observe return to value display.

C. Observer chan LED light.

20. HV ERROR LED's.

A. Attach an HV load to channel 4 of a 1461 module.

B. Push SAVE, then push 2.

C. Observe save successful message.

D. Set this channel to a high target voltage (1461N -1000V), high ramp rate (1461N 2000V/sec), and minimum trip current (1461N 10 uA).

E. Enable this channel and push HVON (channel should trip).

F. Observe HV ERROR LED's flash and HVON LED's flash.

G. Observe HVON characters slowly flashing in spread sheet display.

H. 1454 - Push SYSTEM, and select 3-line display.

I. 1454 - Observe HVON flashing.

21. 1454 - Module channel tripped jump.

A. 1454 - do the following steps with the unit still tripped form the previous test.

B. 1454 - Observe Module LED containing tripped channel continues flashing after all other channels have finished ramping.

C. 1454 - Push Module button for module which does not contain a tripped channel.

D. 1454 - Observe cursor jump to first channel in that module.

E. 1454 - Push Module button for module that does contain tripped channel.

F. 1454 - Observe cursor jump to first tripped channel in that module.

G. Push RECALL, then push 2.

H. Observe restore in progress, be patient message.

I. Observe restore successful message.

22. Module Jump buttons

A. 1454 - Observe that the 1454 front panel module LED's are on only for occupied slots
B. 1454 - Push *Module 0*, *Module 1*, *Module 2*, or *Module 3* for occupied slots only.
C. 1458 - Push Shift-0 … Shift-9 or Shift-a … Shift-f for occupied slots (0-15) only.
D. Observe channel vs. value display jump to first channel of module (FW feature).
E. 1454- Observe Module LED underscore is ON for selected module.
F. Do steps A through E for all installed modules.
G. 1454 - Push *Module 0*, *Module 1*, *Module 2*, or *Module 3* for unoccupied slots only.
H. 1458 - Push Shift-0 … Shift-9 or Shift-a … Shift-f for unoccupied slots (0-15) only.
I. Observe slot not occupied message.
J. 1454 - Do steps 6 and 7 for all unoccupied slots.
K. 1454 - Turn power off , then move modules from occupied slots and to unoccupied slots.
L. 1454 - Turn power on, observe proper module LED's are lighted.
M. 1454- Push *Module* buttons for newly occupied slots and observe underscore LED's.

23. 1454 - Module channel ramping jump.
A. Set a channel to a high target voltage and low ramp rate.
B. Enable this channel and disable all others.
C. Use display control keys to move this channel off the current display.
D. Push HVON, observe Module LED flash for this module containing this channel.
E. Push the flashing module's button, observe display jump to this channel.

24. Remove powerup save set.
A. Push *SAVE*, then push *6*.
B. Observe return to channel vs. value display.
C. AC Cycle power.
D. 1454 - Observe warning that default module settings are being used.
E. Observe 2 beeps when remote/network flashing stops.
F. Push *SYSTEM*, then SELECT "System Information", then SELECT "Powerup Status".
G. Observe Powerup Status display indicates "Warning" and "Powerup Save File Not Found".

25. Display Up

A. 1454 - Push *DISPLAY UP*
B. 1458 - Push *u* (VT100 Interface).
C. Observe Display selection menu and select the "Module Info vs. Lun" display.
D. Activate Display selection menu again with DISPLAY UP or u.
E. Select "Channel vs. Value" display.

26. Help

A. 1454 - Push *Help*

B. 1458 - Push *h* (VT100 Interface).

C. Observe Help display appear and select a topic.

D. Use help key to return to previous topic.

E. 1454 - Push *ESC* to return to main display

F. 1458 - Push *e* to return to main display.

Annotations

The above tests not only show that each front panel or VT100 key works but that it also operates in the expected fashion. Notice that the user can set, increment, decrement, or delta (increase the current value by a specified amount) a specific value or a selected range of values.

While creating save sets via the SAVE button are quite efficient, restoring those sets take quite a bit longer, this due to the need to read the restore file three times: 1) verify save file checksum, 2) check that current module configuration is the same as the save set being restored and that no values exceed either the module limits or the currently active software limits, and 3) load the values into modules. If steps 1) or 2) fail none of the save set is restored.

No attempt is made to delete old save sets which go with module configurations which are different from the current module configuration.

Enabled channels are indicated by dot next their slot-channel designation in the channel vs. value display. This little indicator is unfortunately sometimes not enough to avoid the trap of having no channels enabled when turning on the HV and seeing the HVON light come on, then observing no measured voltage or current and wondering why.

As stated in other sections the HVOFF always works independent of the front panel key switch mode or remote access operating mode.

Some folks might never know that displays other than the channel vs. value display exist. All but two of the “other” displays are simply a single property for all channels. By the way you can set target voltages in the target voltage display just like in the channel vs. value display. Notice the packed status display (an attempt to put as much status information as possible on the screen) and the module display which shows of module ID information for each install module.

The 1454 module buttons play three different roles. When no channels are ramping or tripped, these buttons jump the cursor and/or display to the first channel of the module selected. When no channels are tripped but some are ramping in the selected module, the cursor/display is jumped to the first ramping channel of that module. When channels are tripped in the selected module, the cursor/display is jumped to the first tripped channel of that module.

The Save and Restore menu have the feature that allows the user to remove the power up save set which permits the user to re-establish module default settings without unplugging modules.

System Menu Driven Features

Overview

This section involves the exercising of features of the 1454/8 which are accessible via the system menu.

Procedures

1454 - Configure for **Manual Testing**.

1454 - Front panel key switch in local.

1458 - Configure for **VT100 Testing**.

1458 - Front panel key switch in remote and initiate VT100 display

All tests begin from the system menu, push SYSTEM.

1. No Action Menu escape time-out.
   A. From system menu SELECT “Serial Port Setup”.
   B. Observe that this menu after 60 seconds reverts to the system menu.
   C. Observe that this menu after 60 seconds reverts to the main display sheet.
   D. 1454 - Observe that the main display goes to the three line display after 60 seconds
2. *0) Activate Three Line Display*
   A. SELECT “Activate Three Line Display”.
   B. 1454 - Observe large font three line display.
   C. 1458 - Observe VT100 display does not change.
3. *1) Lock/Unlock Settings/Limits*
   A. Exercise Settings Lock
      1. SELECT “Lock/Unlock Settings/Limits”.
      2. Observe lock menu indicates both settings and limits unlocked.
      3. SELECT settings and enter Lock PIN of “1234”.
      4. ESC to main menu.
      5. Observe “LOCKED” indicator at top of display sheet.
      6. Push HVON and observe HV turn on (allowed even though locked).
      7. Attempt to change a channel’s target voltage.
      8. Observe menu to enter lock PIN and enter PIN “1234”.
      9. Continue with voltage entry.
      10. Observe “LOCKED” indicator no longer present at top of display sheet.
   B. Exercise Limits Lock
      1. From system menu SELECT “Lock/Unlock Settings/Limits”.
      2. SELECT limits and enter Lock PIN of “5678”.
      3. From system menu SELECT “Software Limits”, then observe “LOCKED” indicator at top of limits display sheet.
      4. Attempt to change a channels limit voltage.
      5. Use ESC to leave PIN entry menu, and ESC to leave value entry.

6. Use ESC to leave SW limits display.
7. From system menu SELECT “Lock/Unlock Settings/Limits”.
8. Observe lock menu indicates settings unlocked and limits locked.
9. SELECT limits and enter PIN “5678”.
10. Observe Lock Menu indication that settings are unlocked.

C. Locks preserved after powerdown and clear locks by unplugging all modules.

1. From system menu SELECT “Lock/Unlock Settings/Limits”.
2. SELECT settings and enter Lock PIN of “5678”.
3. SELECT limits and enter Lock PIN of “5678”.
4. Escape to main display and observe LOCKED indication.
5. Cycle power and after ready observe main display “LOCKED” indication.
6. Push System then SELECT “Software Limits” and observe “LOCKED” indication.
7. Turn power off and unplug all modules.
8. Turn power on and observe no-modules-beep sequence, then turn power off.
9. Turn power on and after ready observe main display does not have “LOCKED” indication.
10. Push System then SELECT “Software Limits” and observe display does not have “LOCKED” indication.

4. *2) Software Limits*

A. In the channel vs. value display note a channel set for zero volts.

B. From the system menu SELECT “Software Limits”.

C. Now in the channel vs. limit display set a limit of 1000 volts for the previously noted channel.

D. ESC to the channel vs. value display.

E. Attempt to set a target voltage in the noted channel greater than 1000 volts.

F. Observe error message at bottom of display.

G. Cycle mainframe power.

H. From the system menu SELECT “Software Limits”.

I. Observe that previously entered limit was restored after powerup.

J. Set limit of noted channel back to maximum and ESC to leave SW limits display.

5. *3) System Defaults*

A. From the system menu SELECT “System Defaults”

B. *0) 1454 - Timeout to 3Line Display*

1. The timeout to 3Line Display option should be “YES” by default.
2. SELECT this option , then SELECT “No”, then SELECT “Save”.
3. ESC to channel vs. value display and observe that display does not timeout after 60 seconds to 3-line display.
4. Restore default setup: Reset this option to “YES” and Save.

C. *1) Remote Password Enabled*

1. This option should be “NO” by default.
2. 1454 - Configure for **VT100 testing**.
3. Put front panel switch in REMOTE.

4. To external serial prompt > "VIEW".
5. Note prompt change to "VIEW".
6. Enter > "HVON"
7. Observe error message.
8. Switch front panel switch back and forth between local and remote hitting ENTER after each switch change, then note prompt has "LOCAL" indication when switched to local.
9. 1454 - Leave front panel switch in LOCAL.
10. 1458 - Leave front panel switch in REMOTE.
11. 1458 - Start up VT100 display with "VT100" command.
12. From the system defaults menu SELECT "Remote Password Enabled"
13. SELECT "YES", then SELECT "Save".
14. 1458 - Go back to command line mode by hitting "q".
15. 1454 - Set front panel key switch to REMOTE.
16. From the VT100 terminal Set password > PASSWORD 1234
17. From the VT100 terminal Repeat to verify> PASSWORD 1234
18. Enter to external serial prompt > "EDIT 1234".
19. Enter > "VIEW".
20. Observe error message.
21. Enter > "VIEW 1234".
22. Observe prompt change to VIEW mode.
23. Enter > "HVON"
24. Observe error message.
25. Enter > "EDIT".
26. Observe error message.
27. Enter > "EDIT 1234".
28. Observe prompt change to EDIT mode.
29. 1454 - Set front panel key switch to LOCAL.
30. 1458 - Start up VT100 display with "VT100" command.
31. From system menu SELECT "Remote Password Enabled".
32. SELECT "NO, then SELECT "Save".
33. 1454 - Set front panel key switch to REMOTE.
34. 1458 - Go back to command line mode by hitting "q".
35. Enter > "VIEW".
36. Observe prompt change to VIEW mode.
37. Enter> "EDIT".
38. Observe prompt change to EDIT mode.

D. STATUS Output is grounded until mainframe ready.

1. Connect Voltmeter to STATUS Output.
2. Turn off mainframe, observe Status Output floating.
3. Turn on mainframe, observe Status Output ground until mainframe ready (remote and network LED's stop flashing).

E. *2) STATUS GND on Panic Off*

1. Connect Voltmeter to STATUS Output.
2. This option should be “Yes” by default.
3. Observe STATUS is +5V when Panic Off is not active.
4. Hit the PANIC Off and observe that this output is 0V.
5. Reset Panic off.
6. SELECT this option, then SELECT “NO”, then SELECT “Save”.
7. Observe STATUS is +5V when Panic Off is active.
8. Reset Panic off.
9. Reset this option to “Yes” (don't forget to SELECT “Save”).

F. *3) STATUS GND on any Trip*

1. Connect Voltmeter to STATUS Output.
2. This option should be “No” by default.
3. Cause a tripped channel
   a) Set an enabled channel’s trip current to its minimum value.
   b) Set the same channel’s ramp rate to its maximum value.
   c) Attach HV load to channel output.
   d) Turn HVON, this channel should trip.
4. Observe STATUS output is 5V even though channel tripped.
5. Reset tripped channel via DISABLE.
6. SELECT this option, then SELECT “YES”, then SELECT “Save”.
7. Cause a tripped channel by enabling channel.
8. Observe STATUS output is 0V.
9. Reset tripped channel and observe STATUS output is 5V.
10. Reset this option to “NO” (don't forget to SELECT “Save”).

G. *4) STATUS GND on HVOFF*

1. Connect Voltmeter to STATUS Output.
2. This option should be “No” by default.
3. Observe STATUS output is 5V with HVON and HVOFF (leave HVON).
4. Set this option to “Yes” (don’t forget to SELECT “Save”).
5. Observe STATUS output is 0V when HVOFF and 5V when HVON.
6. Reset this option to “NO” (don’t forget to SELECT “Save*”).*

H. *5) HVON ACFail, restore HVON*

1. This option should be “No” by default.
2. This option is tested in the Safety Features section.

I. *6) HVOFF on GND MACRO*

1. This option should be “No” by default.
2. Turn HVON, then ground MACRO input.
3. Observe HV does not turn off.
4. Remove ground from MACRO input
5. Set this option to “Yes” (don’t forget to SELECT “Save”).
6. Observe that HV turns off when MACRO input grounded.
7. Note that HV stays off when MACRO input is not grounded.
8. Reset this option to “NO” (don’t forget to SELECT “Save”).

J. 7) *Alt. Setup on GND MACRO*

1. This option should be “No” by default.
2. SELECT this option and observe feature not implemented message.

K. Check that system defaults menu only has “Yes” for 3-line display and STATUS panic off option.

6. *1454- 4) Serial Port Setup* - FW feature

A. SELECT this option and confirm default settings of 9600 baud, 8 data bits, 1 stop bit, no parity and echo on

B. 1454 - Changes to serial setup while session active are disabled..

1. Establish terminal command line session.
2. SELECT baud in serial port menu.
3. Observe message that serial port is active.
4. Discontinue session by entering > QUIT

7. *1454 - 4) Serial Port Setup - try all baud rates*

A. SELECT baud in, then SELECT 19,200, then SELECT “Save”.

B. Change terminal baud rate to 19,200 (don’t forget to SELECT “Save”);

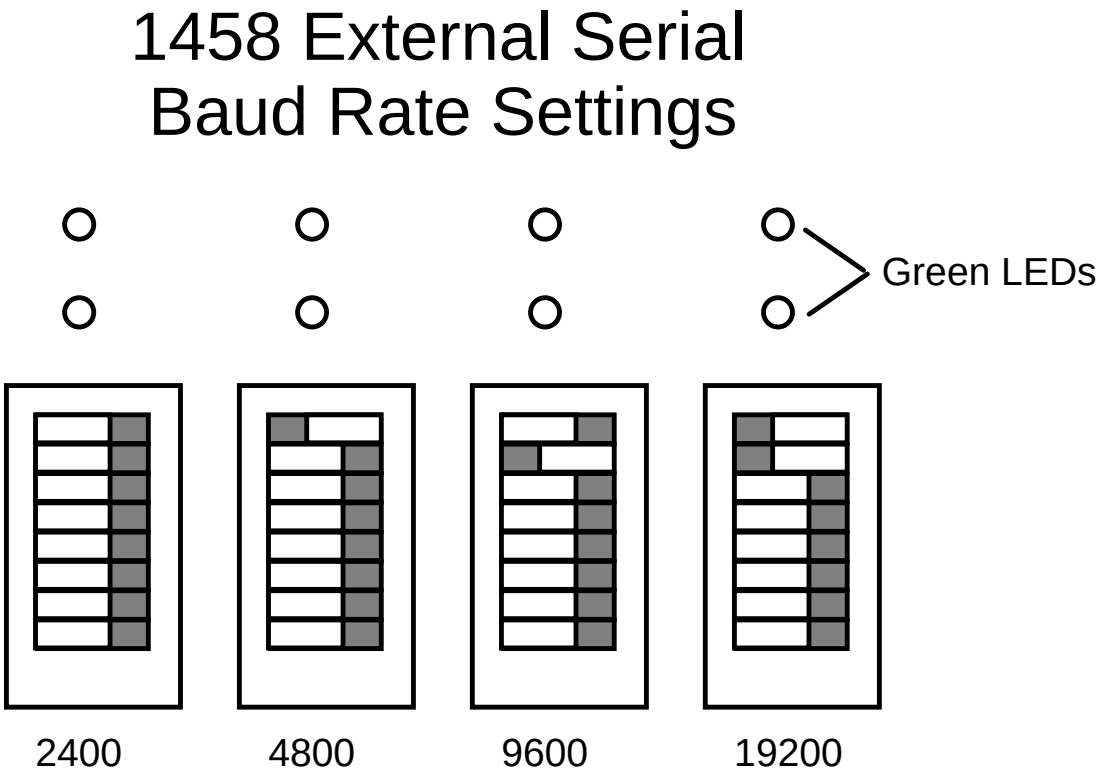
C. Hit return on terminal and observe “0\Enter 1450 to begin>“.

D. Repeat a) - c) for all possible baud rates.

E. Reset baud rate to 9600 (don’t forget to SELECT “Save”).

8. 1458 - Serial Port Setup - try all baud rates.

A. Locate the switch block on the 1450-8 (ISA card with 2 green LEDS above switch block, notice that ARCNET card has one read LED). The 1st and 2nd switches (directly next to the LEDS) control the serial port baud rate. Shown below is are the settings for the 4 possible external serial baud rates:



B. Set the switches for 2400 baud in, then cycle the AC Power and wait for unit to be ready.

C. Change terminal baud rate to 2400.

D. Hit return on terminal and observe "0\Enter 1450 to begin>".

E. Repeat b) - c) for all possible baud rates .

F. Leave set for 9600 baud.

9. *5) Network Setup*

A. SELECT this option and observe ARCNET Node ID.

B. This value should agree with ARCNET card switch settings.

C. Factory default setting should be non-zero and equal to 7.

D. Note: This value only read on powerup.

10. *6) Special Options*

A. SELECT this option, then SELECT 0) Overwrite Module Defaults.

B. Observe error message that Write Inhibited by HW configuration switch.

C. If error message not observed, bad configuration switch?

11. *7) System Information*

A. SELECT this option, and check that Model, HW Revision , HW ECO, Test Date, Test by entries are correct.

B. Check that selecting items in this menu besides item 0) have no effect. (If value entry menu pops up then EEPROM WRITE ENABLED turn it off.)

C. *0) Powerup Status*

1. SELECT this option and observe that all three status lines are normal.
2. Observe BPUPS: line is appropriate for last power down state.
3. Observe that powerup time and update rate are typical for module configuration.

12. *8) 1454 - Screen Contrast*

A. SELECT this option and observe increment/decrement menu.

B. Exercise the INCREMENT and DECREMENT.

1. Press *INCREMENT* repeatedly and observe screen get brighter, then go dark.
2. Press *INCREMENT* repeatedly until proper contrast achieved.
3. Press *DECREMENT* repeatedly and observe screen get darker, then go very bright.
4. Press *DECREMENT* repeatedly until proper contrast achieved.
5. Press *SAVE* to save this setting then hit any key to return to system menu.

C. Check that Escape from Contrast Menu restores previous setting.

1. From system menu SELECT "Screen Contrast".
2. Press INCREMENT until screen slightly brighter.
3. Press ESC and observe previous screen contrast restored.

D. Check that Contrast restored upon powerup.

1. From system menu SELECT "Screen Contrast".
2. Press INCREMENT until screen slightly brighter.
3. Cycle AC power
4. Observe previous screen contrast restored.

13. *9) Reset PANIC OFF*

A. Hit Panic Off

B. 1454 - Observe Panic reset menu and SELECT to reset later.

C. 1454 - Attempt to change demand voltages, observe error message.

D. 1454 - In system menu SELECT this option to reset Panic Off.

E. 1454 - Observe Panic off reset.

F. 1458 - Via VT100 display SELECT this option.

G. 1458 - Observe message warning only front panel reset permitted.

Annotations

After no user input for 60 seconds, all menu's and error messages eventually vanish and the display reverts to the currently active main display sheet. The 1454 will then revert after a additional 60 seconds to the large font "3 line display". The latter 1454 display timeout feature can be disabled in the system defaults menu..

Only the 1454 front panel display supports the large "3 line display".

There two independent types of locks, settings and software limits. Even though settings are locked the user may still start HV generation with HVON. Notice the "LOCKED" indication on the value displays when values are locked and on the software limits display when they are locked. Unlike remote passwords PIN's are only remembered as long as the associated value type is locked; i.e., after a lock is unlocked it can be re-locked with a different PIN. To cover the

situation of a lost or PIN, all locks are removed when the mainframe is powered up with any modules installed. A new module configuration does not clear previously set locks.

Software limits provide a way for the user to set absolute value limits on demand values on a per channel basis. They are preserved between powerups like module settings and saved as part of the named save sets.

System defaults are saved in EEPROM. When changing a system defaults value (in the system defaults submenu), the user must purposely select save to activate any changes.

For 1454's the timeout to the large font "3 line display" is enabled (Yes) by default. The timeout period is 60 seconds.

Remote passwords can be used only during an RS-232 command session to restrict edit access. When remote passwords are enabled, the serial "VIEW" command must be entered with a four digit password number. Subsequent "VIEW" and "EDIT" commands require the same password number be included. The password number remains in effect until remote passwords are disabled either via the system default submenu or by powering up the mainframe with no modules. Unlike a settings lock, a serial session in view mode does not permit the "HVON" command.

Once the mainframe power is turned on the status output is grounded until the mainframe is ready (the remote and network LED's stop flashing) at which time the output level is determined by system defaults.

The mainframe status output can be grounded under any of three possible system default options: 1) when Panic Off is active, 2) when any channel is tripped, or 3) when the HV is off. Whether any of these conditions contribute to the state of the status output is indicated in system default submenu.

The macro input can be configured to turn off HV generation. Once HV is turned off in this manner, a HVON command or button is required to turn HV once the input is no longer grounded.

At some future date, the macro input will support the activation of alternate module setups.

Notice, the serial port configuration cannot be changed once a serial command session has been activated. The ARCNET Node ID is controlled via the switches on the front panel of the ISA bus card. The node ID is read on powerup.

While the external serial port configuration for the 1454 can be changed via the system menu, the 1458 configuration is determined by the first 2 switches on the 1450-8. Only the baud rate can be changed on the 1458. A new switch setting is only recognized after the unit AC power is cycled.

The special options submenu selection currently supports the overwriting of module defaults for 1451's only. This feature is only available if the 1450-1 configuration switch, EEPROM write enable is active.

The system information submenu should have the correct data. Notice the op hours entry which is the total operational hours for the mainframe (maintained as small data file on the mainframe D: drive). The powerup status selection leads to a display which provides further information on the powerup state of the mainframe.

Selecting screen contrast allows the user to adjust the contract using the increment or decrement keys or by selecting these items in the menu. The setting rolls over from maximum contrast to minimum contrast. The set value only takes effect if the user selects save and of course the setting is maintained after a power down.

In keeping with the concept that the Panic Off condition is a local feature, select the panic off reset option is effective when the front panel switch is in local.

VT100 Monitoring - Access Modes

Procedures

Configure for **VT100 Testing**.

Type “q” to exit to prompt 1\edit\1450>.

1. Test Local mode.
   A. Set front panel switch to LOCAL.
   B. Hit <CR> and observe prompt to be “3\local\view\1450>“.
   C. Enter “VT100” to begin VT100 terminal session.
   D. Observe error message response indicating front panel key in local (FW Ver. >= 2.5).
2. Test VIEW command.
   A. Set front panel switch to remote and enter “hi” followed by <CR>.
   B. Observe prompt to be “1\edit\1450>“ and enter VIEW.
   C. Observe prompt to be “2\view\1450>“.
   D. Enter “VT100” to begin VT100 terminal session.
   E. Observe “VIEW” indication at top of display.
   F. Attempt changes of settings, locks, and system defaults, as itemized above.
   G. Hit “q” to quit VT100 terminal session.
   H. Observe prompt to be “2\view\1450>“.
   I. Enter “EDIT” and observe prompt to be “1\edit\1450>“.

3. Test VT100 VIEW command.
   A. Set front panel switch to remote and hit <CR>.
   B. Observe prompt to be “1\edit\1450>“.
   C. Enter “VT100 VIEW” to begin VT100 terminal session.
   D. Observe “VIEW” indication at top of display.
   E. Attempt changes of settings, locks, and system defaults, as itemized above.
   F. Hit “q” to quit VT100 terminal session.
   G. Observe prompt to be “1\edit\1450>“.
4. Test VT100 termination by switch to local (FW Ver. >= 2.5).
   A. Set front panel switch to remote and hit <CR>.
   B. Observe prompt to be “1\edit\1450>“.
   C. Enter “VT100” to begin VT100 terminal session.
   D. Set front panel switch to local and observe VT100 display terminate.
5. Test Panic Off
   A. Set front panel switch to remote and hit <CR>.
   B. Observe prompt to be “1\edit\1450>“
   C. Enter “VT100” to begin VT100 terminal session.
   D. Observe “EDIT” indication at top of display.
   E. Hit front panel panic button.
   F. Observe red panic LED’s and “PANIC” indication at top of display.
   G. Attempt to clear panic via *HVOFF* button and observe error message.
   H. Set front panel switch to LOCAL and observe VT100 display terminate..
   I. Clear panic via HVOFF button then hit <CR> and observe prompt “3\local\view\1450>”.
   J. Set front panel switch to remote then hit <CR> and observe prompt “1\edit\1450>”.

Annotations

For a 1458 mainframe in particular, it is desirable to run a VT100 terminal as simply a monitor with no ability to edit settings or turn on HV while allowing remote ARCNET commands with full access. The above tests illustrate the use of the VIEW and VT100 VIEW commands to support this feature.

The serial command prompt and VT100 display indicates which mode of four is active either “EDIT”, “LOCAL”, “VIEW’, or “PANIC”. The last three modes prohibit setting changes and requests to turn on HV and they are prioritized by their importance (“PANIC”, “LOCAL”, then “VIEW”).

Starting with version 2.50 the VT100 full screen display can only be activated when the key switch is in remote. Also, when the VT100 is active and the key is switch to local this display is terminated.

Remote Interface Testing

RS-232, ARCNET, Ethernet Setup

The following section documents **testing of a 1454/8 HV mainframe via the RS-232, ARCNET, and Ethernet remote interface** using the test software, hvtest.exe and bsdtest.exe. The following hardware configuration is required to use this program most efficiently. Although optional items are followed by a "*", testing without these items is awkward and less efficient.

1. 386 PC
   A. hvtest.exe installed on hard drive.
   B. An available serial port and a null modem cable.
   C. An installed ARCNET interface (SMC PC600WS) .
   D. An installed Ethernet interface (Mitron Ethernet ISA card LX-2000JL).
   E. ARCNET connections require 2 - BNC T's, 2 - 90-Ohm loads, and RG-62 cable.
   F. Ethernet connections require 2 - BNC T's, 2 - 50-Ohm loads, and RD-58 cable.
   G. An available printer port for TTL adapter. (*)
2. 1454 or 1458 HV mainframe (front panel key switch in remote).
3. At least 1 1461N Module installed in slot 0 of HV mainframe (No 1461P's).
4. A single HV load (~ 1Mohm) for Channel 0 of the 1461N module in slot 0.
5. A TTL controlled AC power switch box to control the mainframe's AC power. (*)
6. A PC Parallel Port TTL I/O adapter. (*)
   A. HV mainframe Power controlled via TTL I/0 Line 0 (AutoFeed , Pin 14, Offset 2, Bit 1).
   B. HV mainframe Macro connected to TTL I/0 Line 1 (Select , Pin 17, Offset 2, Bit 3).
   C. HV mainframe Status connected to. TTL I/0 Line 2 (PaperEnd, Pin 12, Offset 1, Bit 5).

Remote Testing Procedures

The following lists step by step procedures for using HVTEST to do remote testing. See following sections for more details on HVTEST operation and features.

1. ARCNET testing with HVTEST.

A. Check hvtest.exe's default configuration for ARCNET interface card (interrupt and port) are same as that installed in test PC by entering "HVTEST ?" to the DOS prompt.

B. To the DOS prompt enter "HVTEST L2 R7". More command line options may be needed to accommodate an ARCNET card with non-default jumper settings.

C. With mainframe powered up in REMOTE and ARCNET cable connected, enter "hi" to the hvtest prompt ">".

D. Mainframe should respond with "Hi How are you?"

E. To the hvtest prompt ">" enter "~t ?" to use display information to confirm test setup and the possible need for addition test script command line options.

F. To the hvtest prompt ">" enter "~t" to begin automated testing. More test options may be required to accommodate special test conditions suggested in the previous step.

2. Ethernet (1450-ET) testing with BSDTEST.

A. Check bsdtest.exe's default configuration for test PC Ethernet interface card (interrupt and port) are same as that installed in test PC by entering "BSDTEST ?" to the DOS prompt.

B. To the DOS prompt enter "BSDTEST ". More command line options may be needed to accommodate an Ethernet card with non-default jumper settings or an HV mainframe with non-default IP address and BSD port settings. (Use serial connection and ENET command to check these if connection fails).

C. With mainframe powered up in REMOTE and Ethernet cable connected, enter "hi" to the bsdtest prompt ">".

D. Mainframe should respond with "Hi How are you?"

E. To the bsdtest prompt ">" enter "~t ?" to use display information to confirm test setup and the possible need for addition test script command line options.

F. To the bsdtest prompt ">" enter "~t" to begin automated testing. More test options may be required to accommodate special test conditions suggested in the previous step.

3. RS-232 testing with HVTEST.

A. Check hvtest.exe's default configuration for serial com port are same as to be used for communicating to the mainframe by entering "HVTEST ?" to the DOS prompt.

B. To the DOS prompt enter "HVTEST " or "HVTEST C#". Where "#" is a non-default com port number.

C. With mainframe powered up in REMOTE and the serial cable connected, hit a <ENTER>.

D. Mainframe should respond with "0\Enter "1450" to begin>"

E. To the hvtest prompt ">" enter 1450 and observe the mainframe responding prompt "1\edit\1450>"

F. Enter "hi" to the hvtest prompt ">".

G. Mainframe should respond with "Hi How are you?"

H. "~t ?" to use display information to confirm test setup and the possible need for addition test script command line options.

I. To the hvtest prompt ">" enter "~t" to begin automated testing. More test options may be required to accommodate special test conditions suggested in the previous step.

4. ARCNET testing - no modules.

A. Unplug all modules and powerup mainframe and observe NETWORK/REMOTE LED's flashing and repeated two beeps.

B. Repeat steps B through D of item I.

C. To the hvtest prompt ">" enter "sysinfo" and observe proper information.

D. To the hvtest prompt ">" enter hvstatus and observe no modules error message.

5. RS-232 testing - no modules.

A. Unplug all modules and powerup mainframe and observe NETWORK/REMOTE LED's flashing and repeated two beeps.

B. Repeat steps B through G of item III.

C. To the hvtest prompt ">" enter "sysinfo" and observe proper information.

D. To the hvtest prompt ">" enter hvstatus and observe no modules error message.

HVTEST and BSDTEST Operation

The hvtest and bsdtest programs have a number of command line options. These permit the user to specify interface port addresses and/or interrupts which are different from program defaults as well activate a particular communications port. A list of all command line options can be viewed by entering “HVTEST ?” or “BSDTEST ?to the DOS prompt.

To activate an RS-232 connection on PC COM 2 enter “HVTEST C2”.

To activate an ARCNET connection (for an ARCNET interface installed with the proper address and interrupt defaults) enter “HVTEST L2 R7”. This command sets the PC ARCNET node ID equal to 2 and will later send ARCNET packets to ARCNET node ID 7 (Default HV mainframe node ID).

To activate an Ethernet connection enter “BSDTEST”.

Once the activated, hvtest and bsdtest present a simple prompt “>“. Further keyboard entry by the user is echoed by the program until the user hits ENTER. If the first character of the entered string is not a “~” or “@” then the string is sent to the HV mainframe via the previously designated communications port.

For a complete hardware setup, the standard 1454/8 remote interface test is started by entering “~T” to the hvtest prompt. Use “~T ?” obtain a listing of options for the test script thread T.

Command Procedures

Entered strings beginning with a “@” specify a command file be used as source for commands. For example,”> @myfile.cmd” sends command strings from the file, myfile.cmd. This command file is a text file created by the user using (for example) the DOS edit command.

Text strings in a command file which begin with a “:” indicate a command file control operation. Currently supported command file control operations are “:GOTO XXXX”, “:LABEL XXXX”, “:LOOP N”, “:LOOP”, “:ENDLOOP”, and “:END”. The string “XXXX” indicates an arbitrary text word used with the GOTO and LABEL control operations. The LOOP control operation contains a loop counter “N”. If “N” is not specified or is equal to 0 the procedure loops forever. The “:END” operation terminates the command procedure and also command file looping via “>@ myfile.cmd n”. The “n” this case indicates the number of complete loops to be executed. A value of zero for “n” loops forever.

Script Threads

Entered strings beginning with a “~” activate script threads. Currently the only supported script thread is the “T” or test script thread which one begins with “ ~T” at the hvtest or bsdtest prompt (“>”).

A number of test script thread options are supported some of which allow the user to execute the test script without the optional test hardware listed earlier. To view a complete list of these options enter
“~T ?“. For typical mainframe testing with a complete hardware test setup, no options are required. Listed below are currently supported test script thread options.

A --- AC power controlled manually. (Def: Automated)
L# -- Loop Test script. exit on failure. (# - number of loops, 0 = forever)
P# -- TTL Port designation. (Def: Lpt2)
0 - LptMDA -> 0x03BC
1 - Lpt1 -> 0x0278
2 - Lpt2 -> 0x0378

3 - LptNone -> TTL I/O done manually
SK -- Skip Panic Off Status Check.
U# -- Only execute loop portion of script. (# - number of loops)
V# -- Script Verification Level. (Def: VerifySections)
0 - VerifyNone 1 - VerifySections
2 - VerifyCommands 3 - VerifyControls
4 - VerifyComments 5 - VerifyResponses
6 - VerifyReceives 7 - VerifyAll
X --- Exercise TTL I/O port only.

The Test Script Thread

The test script functions (listed in a later section) execute as a separate thread within a multi-threaded program environment provided by the hvtest executable. The multi-threaded nature of hvtest and bsdtest support the immediate termination of a script thread by low-level test functions (like tsend(), tmatch(), etc.) without terminating the hvtest or bsdtest programs. This allows script functions to call test functions without checking their return status after each call which greatly simplifies script functions.

The most frequently used test functions are tsend() and tmatch() whose operation we will discuss briefly in the following paragraphs. Other test functions will eventually be discussed as an appendix to this document.

Whether any test function (which detects an error) terminates the script thread or simply returns an error status code to the calling script function is determined by the test function tmode(). The test function tmode() can be called with the arguments ExitOnError, ExitOnWarning, or ReturnStatus. Typically the mode, ExitOnError, is active for the Test script function.

Important Test Functions

tsend()

The tsend() test function transmits the argument string to HV mainframe and waits for a return message. (In the case of RS-232, the character ">" indicates the end of a message, while for ARCNET and Ethernet the lack of continuation character "C" in the response packet indicates the end of a message. See the 1454/8 User's Guide.)

The mainframe return message has two parts the command response and return prompt. The tsend() only "checks" the return prompt if the tmatch() function has *not* been called prior to the last tsend() call. When tsend() checks the return prompt, it only checks for a "normal prompt" or a normal return status (between 1 and 20). A non-normal prompt returns status causes tsend() to issue an error.

tmatch()

The tmatch() function checks both the return prompt and a single line of the returned command response based on input arguments. The first argument is the prompt string to be matched while the second argument is a "format string" to be matched to a single line of the command response. The format string makes use of some special characters (listed below) to allow for generalized string matching. Additional tmatch() arguments (pointers to strings) should be supplied when the format string specifies substitution regions.

The tmatch() breaks up a command response line into substrings or tokens using the space character as separators. Listed below are the special characters and their function in tmatch() format strings.

%	Ignore further tokens.
*	Match any character in this token.

?	Match any character at this location in this token.
^	Encloses substitution regions.

Using the specifier, NORMAL_PROMPT as the first argument to tmatch() results in only checking the prompt for a normal status. The IGNORE_RESPONSE specifier for the second argument of tmatch() causes tmatch() to ignore a single line of the command response.

tverify()

The tverify() function allows the user to set the amount of information displayed as the test script runs. Listed below are the possible verification levels. Notice that the Test script thread has a user option to set the verification level when the script thread is started.

0 - VerifyNone	1 - VerifySections
2 - VerifyCommands	3 - VerifyControls
4 - VerifyComments	5 - VerifyResponses
6 - VerifyReceives	7 - VerifyAll

All screen output from test functions and script functions should be proceeded by an identifying prefix. A listing of possible prefixes and their interpretation follows:

:	Match Prompt
#	Match Response
$	Send
$NW	Send No Wait
!	Comment
!D	Dump (After timeout)
!S	Section
!M	Operator Message
#R	Receive Response
<	Receive Response
~	Control
~ERROR	Error
~EXIT	Exit
~FAIL	Fail

Test Script Function Listing

The following pages list all of the script functions called as part of the test script thread. The Test script thread typically begins with the execution of the script function, script_mfcmd().

```
// MFCMD - Exercise each command and its features. */
int script_mfcmd(int skip_panic_check)
{
char mf_config0[HEX4LEN + 1], mf_config1[HEX4LEN+1],mf_model[HEX4LEN + 1],
     set_summary[HEX4LEN+1], mf_activity0[HEX4LEN+1], mf_activity1[HEX4LEN+1],
     host_activity[HEX4LEN+1], measured_summary0[HEX4LEN+1],
measured_summary1[HEX4LEN+1],
     tmp_hex4[HEX4LEN+1];
char buf[MAX_MESSAGE_LEN+1], sw_limit_buf[21];
int i, k, number_of_modules, number_of_properties;
int yes_or_no;
COM_PORTS com_port;
float required_fw_version;
float mf_fw_version;

   required_fw_version = (float) 2.34;
   tversion("Mfcmd - V1.52");
   tmessage(          " Required Setup:");
   IW_sprintf(buf,    "  1454 or 1458 HV mainframe with FW Version >= %.3f",
              required_fw_version);
   tmessage(          buf);
   tmessage(          "  At least 1 1461N or 1461P Module installed in slot 0.");
   tmessage(          "  Channel 0 of slot 0 should have an HV load.");
   if (0 != tport_base_address()) {
      IW_sprintf(buf, "  Parallel Port TTL I/O Adapter at address %X (HEX)",
                 tport_base_address());
      tmessage(        buf);
   } else {
      tmessage(       "  TLL I/O functions provided manually");
   }
   if (Automated == tfetch_ac_power_mode()) {
       tmessage(      "  HV mainframe Power TTL controlled via...TTL I/0 Line
0.");
   } else {
       tmessage(      "  HV mainframe Power controlled..........Manually.");
   }
   tmessage(          "  HV mainframe Macro connected to.........TTL I/0 Line
1.");
   tmessage(          "  HV mainframe Status connected to........TTL I/0 Line
2.");
   tmessage(          "  HV mainframe front panel key switch in REMOTE.");
   tmessage(          "  HV mainframe PANIC OFF should be clear.");
   tmessage(          " Check connections enter Y to continue (Def:N)>");
   tyes_or_no(&yes_or_no, ALL_TIME, FALSE);
   if (!yes_or_no) {
      tfail("Configuration not confirmed.");
   }
   tcom_port(&com_port);

// debug
//   tsend(                  "ll");
//   tmatch(NORMAL_PROMPT,   "ll %");
//   number_of_modules = tfetch_last_matched_token_count() - 1;
//   goto testing;
//

   tsection("Mfcmd - Initial Powerup");
   script_accycle();
   script_init();

//
//************************************************************
// Begin portion of script which requires operator intervention.
//************************************************************/
// Detection of PANIC OFF condition
// Note the current configuration summary number
   tsection("Mfcmd - Panic Off Status Output.");
```

```
   tsend(                    "sysdef 000B");
   tmatch(NORMAL_PROMPT, "sysdef 000B");
   tsend(                    "config");
   tmatch(NORMAL_PROMPT, "config 0??? * * * *");
   tcheck_port_in(StatusPort, TTL_ON);
   if (skip_panic_check) {
      tmessage("Panic Off Status ground enabled/disabled skipped.");
      tmessage(">> Operator Input not required. <<");
   } else {
      tsection("Mfcmd - Panic Off Status ground enabled.");
      tsend(                "gs");
      tmatch(NORMAL_PROMPT, "gs * * * * *");
      tcopy_matched_token(mf_config0, 3, HEX4LEN);
      tmessage("Hit PANIC OFF, then any key to continue.");
      twait_for_any_key(ALL_TIME);
   // New config summary number should be incremented.
      tcheck_port_in(StatusPort, TTL_OFF);
      tinc_hex4(mf_config0);
      tsend(                    "gs");
      tmatch(NORMAL_PROMPT,  "gs * * ^mf_config0^ * *", mf_config0);
      tsend(                    "config");
      tmatch(NORMAL_PROMPT,  "config 8??? * * * *");
      tmessage("Clear PANIC OFF:");
      tmessage("  Switch to LOCAL.");
      tmessage("  Hit HVOFF.");
      tmessage("  Switch back to REMOTE.");
      tmessage("  Hit any key when complete.");
      twait_for_any_key(ALL_TIME);
   // New config summary number should be incremented.
      tsend(                     "gs");
      if (com_port == ComSerial) {
         tmatch("1\\EDIT\\1450>", "gs * * * * *");
      } else {
         tmatch("    1", "gs * * * * *");
      }
      tcopy_matched_token(mf_config1, 3, HEX4LEN);
      if (mf_config1 == mf_config0) {
         tfail("Configuration failed to change.");
      }
      tsend(                 "config");
      tmatch(NORMAL_PROMPT,  "config 0??? * * * *");
      tsection("Mfcmd - Panic Off Status ground disabled.");
   // Turn off Panic -> status ground via sysdef and check
      tsend(                 "sysdef 0003");
      tmatch(NORMAL_PROMPT,  "sysdef 0003");
      tmessage("Hit PANIC OFF, then any key to continue.");
      twait_for_any_key(ALL_TIME);
      tcheck_port_in(StatusPort, TTL_ON);
      tmessage("Clear PANIC OFF:");
      tmessage("  Switch to LOCAL.");
      tmessage("  Hit HVOFF.");
      tmessage("  Switch back to REMOTE.");
      tmessage("  Hit any key when complete.");
      twait_for_any_key(ALL_TIME);
      tsend(                   "sysdef 000B");
      tmatch(NORMAL_PROMPT,   "sysdef 000B");
      tmessage(">> Operator Input no longer required. <<");
   }
//
//*********************************************************
// The rest of script does not require operator input
//*********************************************************

   script_inicfg(required_fw_version, mf_model, &mf_fw_version);

   tsection("Mfcmd");
   tmode(ExitOnError);

// Use 6th save set as script default
   tsend(                  "save 5");
   tmatch(NORMAL_PROMPT,   "save 5 complete");
//
// Determine the number of modules by counting tokens returned by "ll"
   tsend(                  "ll");
   tmatch(NORMAL_PROMPT,   "ll %");
```

```
   number_of_modules = tfetch_last_matched_token_count() - 1;

//
//
//*************************************************************
// <ATTR>
//*************************************************************
   tsection("Mfcmd - ATTR");
   tsend(                  "attr l0 dv");
   tmatch(NORMAL_PROMPT,   "attr l0 dv * * * * * *");
//
//*************************************************************
// <CONFIG>
// ************************************************************
   tsection("Mfcmd - CONFIG");
   tsend(                  "config");
// "?" is to support any baud rate
   tmatch(NORMAL_PROMPT,   "config 0174 000? 0007 * 000B");
   tsend(                  "hvon");
   tmatch(NORMAL_PROMPT,   "hvon");
   tsend(                  "config");
   tmatch(NORMAL_PROMPT,   "config 2174 000? 0007 * 000B");
   tsend(                  "hvoff");
   tmatch(NORMAL_PROMPT,   "hvoff");
   tsend(                  "lock settings 1234");
   tmatch(NORMAL_PROMPT,   "lock settings locked");
   tsend(                  "config");
   tmatch(NORMAL_PROMPT,   "config 0175 000? 0007 * 000B");
   tsend(                  "lock swlimits 5678");
   tmatch(NORMAL_PROMPT,   "lock swlimits locked");
   tsend(                  "config");
   tmatch(NORMAL_PROMPT,   "config 0177 000? 0007 * 000B");
   tsend(                  "unlock settings 1234");
   tmatch(NORMAL_PROMPT,   "unlock settings unlocked");
   tsend(                  "config");
   tmatch(NORMAL_PROMPT,   "config 0176 000? 0007 * 000B");
   tsend(                  "unlock swlimits 5678");
   tmatch(NORMAL_PROMPT,   "unlock swlimits unlocked");
   tsend(                  "config");
   tmatch(NORMAL_PROMPT,   "config 0174 000? 0007 * 000B");
//
//*************************************************************
// <DATE>
//*************************************************************
   tsection("Mfcmd - DATE");
   tsend(                  "date");
// Used this match format instead of single string to get around
// an apparent C string processing "feature".
   tmatch(NORMAL_PROMPT,   "date " "*" "-???" "-????");
//
//*************************************************************
// <DMP>
//*************************************************************
   tsection("Mfcmd - DMP");
   tsend(                  "prop l0");
   tmatch(NORMAL_PROMPT,   "prop l0 %");
   number_of_properties = tfetch_last_matched_token_count() - 2;
   tsend(                  "dmp l0.0");
   tmatch(NORMAL_PROMPT,   "dmp l0.0 %");
   if (number_of_properties != (tfetch_last_matched_token_count() - 2)) {
      tfail("Property count mismatch.");
   }
//

#if 0
//*************************************************************
// <ECHO>
//*************************************************************
   tget(PropTestWithEchoOff, &i);
   if (!i) {
      tsection("Mfcmd - ECHO");
      tsend(                  "echo off");
      tmatch(NORMAL_PROMPT,   "echo off");
      texpect_serial_echo(FALSE);
      tsend(                  "dmp l0.0");
```

```
      tmatch(NORMAL_PROMPT,   "dmp l0.0 %");
      tsend(                  "echo on");
      tmatch(NORMAL_PROMPT,   "echo on");
      texpect_serial_echo(TRUE);
   }
//
#endif

//*********************************************************
// <EDIT> <VIEW> Serial Session only
//*********************************************************
   tsection("Mfcmd - EDIT, VIEW");
   if (com_port != ComSerial) {
      tsend(               "edit");
      tmatch("  220",      "ERROR - #220 |Serial Command Only|");
      tsend(               "view");
      tmatch("  220",      "ERROR - #220 |Serial Command Only|");
   } else {
      tsend(                     "edit");
      tmatch("1\\edit\\1450>",   "edit");
      tsend(                     "view");
      tmatch("2\\view\\1450>",   "view");
      tsend(                     "hvon");
      tmatch("176\\view\\1450>", "ERROR - #176 |Session is in View Mode|");
      tsend(                     "ld l0.0 dv 0");
      tmatch("176\\view\\1450>", "ERROR - #176 |Session is in View Mode|");
      tsend(                     "sysdef 000B");
      tmatch("176\\view\\1450>", "ERROR - #176 |Session is in View Mode|");
      tsend(                     "lock settings 1234");
      tmatch("176\\view\\1450>", "ERROR - #176 |Session is in View Mode|");
      tsend(                     "edit");
      tmatch("1\\edit\\1450>",   "edit");
   }
//
//*********************************************************
// <GS> <HVON> <HVOFF> <HVSTATUS>
//*********************************************************
//
   tsection("Mfcmd - GS");
   tsend(                  "gs");
   tmatch(NORMAL_PROMPT,   "gs * * * * *");
   tcopy_matched_token( set_summary,   2, HEX4LEN);
   tcopy_matched_token( mf_config0,     3, HEX4LEN);
   tcopy_matched_token( mf_activity0,  4, HEX4LEN);
   tcopy_matched_token( host_activity, 5, HEX4LEN);
   tinc_hex4(host_activity);
// Give mainframe time to update
   tdelay(5);
//      HOST_ACTIVE should increase by one
   tsend(                  "gs");
   tmatch(NORMAL_PROMPT,   "gs * ^SET_SUMMARY^ ^mf_config0^ * ^HOST_ACTIVITY^",
                                set_summary,  mf_config0,    host_activity);
   tcopy_matched_token( measured_summary0,  1, HEX4LEN);
   tcopy_matched_token( mf_activity1,       4, HEX4LEN);
// Only Host activity should change and possibly Meas_sum0 from their previous
values.
   if (0 == IW_stricmp(mf_activity0, mf_activity1)) {
      tfail("Mf Activity summary failed to change");
   }
   if (Lun_Map[0] == Module1461N) {
      tsend(                  "ld l0.0 dv -100.0");
   } else {
      tsend(                  "ld l0.0 dv 100.0");
   }
   tmatch(NORMAL_PROMPT,   "ld l0.0 dv *");
   tsend(                  "ld l0.0 ce 1");
   tmatch(NORMAL_PROMPT,   "ld l0.0 ce *");
   tsend(                  "gs");
   tinc_hex4(host_activity);

#ifdef BUGFIX_0
   tinc_hex4(set_summary); // Once for load of dv
   tinc_hex4(set_summary); // Again for load of ce
   tmatch(NORMAL_PROMPT,   "gs * ^SET_SUMMMARY^ ^mf_config0^ * ^HOST_ACTIVITY^",
                                set_summary,   mf_config0,    host_activity);
```

```
#else
   tmatch(NORMAL_PROMPT,   "gs * * ^mf_config0^ * ^HOST_ACTIVITY^",
                                 mf_config0,    host_activity);
   tcopy_matched_token(tmp_hex4,  2, HEX4LEN);
   if (0 == IW_stricmp(tmp_hex4, set_summary)) {
      tfail("Set Summary failed to change");
   }
#endif
   tsend(                  "hvon");
   tmatch(NORMAL_PROMPT,   "hvon");
   tdelay(2);
   tsend(                  "hvstatus");
   tmatch(NORMAL_PROMPT,   "hvstatus hvon");
   tinc_hex4(host_activity);
   tinc_hex4(mf_config0);    //hvon is a config change
   tsend(                  "gs");
#ifdef BUGFIX_0
   tmatch(NORMAL_PROMPT,   "gs * ^SET_SUMMARY^ ^mf_config0^ * ^HOST_ACTIVITY^",
                                 set_summary,  mf_config0,    host_activity);
#else
   tmatch(NORMAL_PROMPT,   "gs * * ^mf_config0^ * ^HOST_ACTIVITY^",
                                   mf_config0,    host_activity);
#endif
   tcopy_matched_token( measured_summary1,  1, HEX4LEN);
   if (0 == IW_stricmp(measured_summary0, measured_summary1)) {
      tfail("Measured summary value failed to change.");
   }
   tsend(                  "hvoff");
   tmatch(NORMAL_PROMPT,   "hvoff");
   tsend(                  "hvstatus");
   tmatch(NORMAL_PROMPT,   "hvstatus hvoff");
   tsend(                  "ld l0.0 dv 0.0");
   tsend(                  "ld l0.0 ce 0");
//
//*************************************************************
// <ID>
//*************************************************************
   tsection("Mfcmd - ID");
   tsend(                  "id l0");
// Allow for additional ID tokens
   tmatch(NORMAL_PROMPT,   "id l0 * * * * * * * * *");
//
//*************************************************************
// <IMOFF>
//*************************************************************
   tsection("Mfcmd - IMOFF");
   tsend(                  "hvon");
   tdelay(5);
   tsend(                  "hvstatus");
   tmatch(NORMAL_PROMPT,   "hvstatus hvon");
   tsend(                  "imoff");
   tmatch(NORMAL_PROMPT,   "imoff");
   tsend(                  "hvstatus");
   tmatch(NORMAL_PROMPT,   "hvstatus hvoff");
//
//*************************************************************
// <LD> <RC>
//*************************************************************
   tsection("Mfcmd - LD, RC");
   if (Lun_Map[0] == Module1461N) {
      tsend(                  "ld l0.1 dv -50.0");
      tmatch(NORMAL_PROMPT,   "ld l0.1 dv -50.?");
      tsend(                  "rc l0.1 dv");
      tmatch(NORMAL_PROMPT,   "rc l0.1 dv -50.?");

      tsend(                  "ld l0 dv -100.0 -101.0 -102.0 -103.0"
                                     " -104.0 -105.0 -106.0 -107.0"
                                     " -108.0 -109.0 -110.0 -111.0");
      tmatch(NORMAL_PROMPT,   "ld l0 dv -100.? -101.? -102.? -103.?"
                                     " -104.? -105.? -106.? -107.?"
                                     " -108.? -109.? -110.? -111.?");
      tsend(                  "rc l0 dv");
      tmatch(NORMAL_PROMPT,   "rc l0 dv -100.? -101.? -102.? -103.?"
                                     " -104.? -105.? -106.? -107.?"
                                     " -108.? -109.? -110.? -111.?");
```

```
   } else {
      tsend(                  "ld l0.1 dv 50.0");
      tmatch(NORMAL_PROMPT,   "ld l0.1 dv 50.?");
      tsend(                  "rc l0.1 dv");
      tmatch(NORMAL_PROMPT,   "rc l0.1 dv 50.?");

      tsend(                  "ld l0 dv 100.0 101.0 102.0 103.0"
                                      " 104.0 105.0 106.0 107.0"
                                      " 108.0 109.0 110.0 111.0");
      tmatch(NORMAL_PROMPT,   "ld l0 dv 100.? 101.? 102.? 103.?"
                                      " 104.? 105.? 106.? 107.?"
                                      " 108.? 109.? 110.? 111.?");
      tsend(                  "rc l0 dv");
      tmatch(NORMAL_PROMPT,   "rc l0 dv 100.? 101.? 102.? 103.?"
                                      " 104.? 105.? 106.? 107.?"
                                      " 108.? 109.? 110.? 111.?");
   }
   tsend(                 "ld l0 dv 0 0 0 0 0 0 0 0 0 0 0 0");
//
//*********************************************************
// <LL>
//*********************************************************
   tsection("Mfcmd - LL");
   tsend(                 "ll");
   tmatch(NORMAL_PROMPT,  "ll %");
   if (number_of_modules != tfetch_last_matched_token_count() - 1) {
      tfail("ll failed to return proper token count");
   }
//
//*********************************************************
// <LM> <RM>
//*********************************************************
//
   tsection("Mfcmd - LM, RM");
// Save limit to be restored later
   tsend(                 "rm l0.0 dv");
   tmatch(NORMAL_PROMPT,  "rm l0.0 dv *");
   tcopy_matched_token(sw_limit_buf,  3, 20);
//
   tsend(                 "lm l0.1 dv 500.0");
   if (Lun_Map[0] == Module1461N) {
      tmatch(NORMAL_PROMPT,  "lm l0.1 dv -500.0");
   } else {
      tmatch(NORMAL_PROMPT,  "lm l0.1 dv 500.0");
   }
   tsend(                 "rm l0.1 dv");
   if (Lun_Map[0] == Module1461N) {
      tmatch(NORMAL_PROMPT,  "rm l0.1 dv -500.0");
   } else {
      tmatch(NORMAL_PROMPT,  "rm l0.1 dv 500.0");
   }
   tsend(                 "lm l0 dv 1000.0 1010.0 1020.0 1030.0"
                                   " 1040.0 1050.0 1060.0 1070.0"
                                   " 1080.0 1090.0 1100.0 1110.0");
   if (Lun_Map[0] == Module1461N) {
      tmatch(NORMAL_PROMPT,  "lm l0 dv -1000.0 -1010.0 -1020.0 -1030.0"
                                      " -1040.0 -1050.0 -1060.0 -1070.0"
                                      " -1080.0 -1090.0 -1100.0 -1110.0");
      tsend(                 "rm l0 dv");
      tmatch(NORMAL_PROMPT,  "rm l0 dv -1000.0 -1010.0 -1020.0 -1030.0"
                                      " -1040.0 -1050.0 -1060.0 -1070.0"
                                      " -1080.0 -1090.0 -1100.0 -1110.0");
      tsend(                    "ld l0.3 dv -2000.0");
   } else {
      tmatch(NORMAL_PROMPT,  "lm l0 dv 1000.0 1010.0 1020.0 1030.0"
                                      " 1040.0 1050.0 1060.0 1070.0"
                                      " 1080.0 1090.0 1100.0 1110.0");
      tsend(                 "rm l0 dv");
      tmatch(NORMAL_PROMPT,  "rm l0 dv 1000.0 1010.0 1020.0 1030.0"
                                      " 1040.0 1050.0 1060.0 1070.0"
                                      " 1080.0 1090.0 1100.0 1110.0");
      tsend(                    "ld l0.3 dv 2000.0");
   }
   if (com_port == ComSerial) {
      tmatch("159\\EDIT\\1450>", "ERROR - #159 |Value Exceeds SW Limit|");
```

```
   } else {
      tmatch("  159",           "ERROR - #159 |Value Exceeds SW Limit|");
   }
   if (Lun_Map[0] == Module1461N) {
      tsend(                  "ld l0.3 dv -900");
      tmatch(NORMAL_PROMPT,   "ld l0.3 dv -900.?");
   } else {
      tsend(                  "ld l0.3 dv 900");
      tmatch(NORMAL_PROMPT,   "ld l0.3 dv 900.?");
   }
   ttimeout(EXTENDED_TIMEOUT);
   tsend(                  "restore 5");
   tmatch(NORMAL_PROMPT,   "Restore In progress... Please be patient");
   tmatch(NORMAL_PROMPT,   "Restore 5 Complete");
   ttimeout(NORMAL_TIMEOUT);
// Restore does not overwrite limits
   IW_sprintf(buf,         "lm l0 dv %s %s %s %s %s %s %s %s %s %s %s %s",
                                 sw_limit_buf, sw_limit_buf, sw_limit_buf,
                                 sw_limit_buf, sw_limit_buf, sw_limit_buf,
                                 sw_limit_buf, sw_limit_buf, sw_limit_buf,
                                 sw_limit_buf, sw_limit_buf, sw_limit_buf);
   tsend(                  buf);
//
//*********************************************************
// <LOCK> <UNLOCK>
//*********************************************************
   tsection("Mfcmd - Lock");
   tsend(                  "lock settings");
   tmatch(NORMAL_PROMPT,   "lock settings unlocked");
   tsend(                  "lock settings 1234");
   tmatch(NORMAL_PROMPT,   "lock settings locked");
   if (Lun_Map[0] == Module1461N) {
      tsend(                  "ld l0.1 dv -100.0");
   } else {
      tsend(                  "ld l0.1 dv 100.0");
   }
   if (com_port == ComSerial) {
      tmatch("203\\EDIT\\1450>", "ERROR - #203 |Changes to Settings Locked Out|");
   } else {
      tmatch("  203",           "ERROR - #203 |Changes to Settings Locked Out|");
   }
   tsend(                  "hvon");
   tmatch(NORMAL_PROMPT,   "hvon");
   tsend(                  "hvoff");
   tmatch(NORMAL_PROMPT,   "hvoff");
   tsend(                  "lock swlimits");
   tmatch(NORMAL_PROMPT,   "lock swlimits unlocked");
   tsend(                  "lock swlimits 6789");
   tmatch(NORMAL_PROMPT,   "lock swlimits locked");
// Save limit to be restored later
   tsend(                  "rm l0.2 dv");
   tmatch(NORMAL_PROMPT,   "rm l0.2 dv *");
   tcopy_matched_token(sw_limit_buf,  3, 20);
//
   tsend(                     "lm l0.2 dv 1000.");
   if (com_port == ComSerial) {
      tmatch("204\\EDIT\\1450>", "ERROR - #204 |Changes to SW Limits Locked
Out|");
   } else {
      tmatch("  204",           "ERROR - #204 |Changes to SW Limits Locked
Out|");
   }
   tsend(                     "unlock settings 4321");
   if (com_port == ComSerial) {
      tmatch("202\\EDIT\\1450>", "ERROR - #202 |PIN Mismatch|");
   } else {
      tmatch("  202",           "ERROR - #202 |PIN Mismatch|");
   }
   tsend(                  "unlock settings 1234");
   tmatch(NORMAL_PROMPT,   "unlock settings unlocked");
   if (Lun_Map[0] == Module1461N) {
      tsend(                  "ld l0.1 dv -100.0");
   } else {
      tsend(                  "ld l0.1 dv 100.0");
   }
```

```
   tmatch(NORMAL_PROMPT,   "ld l0.1 dv *");
   tsend(                  "ld l0.1 dv 0");
   tsend(                     "unlock swlimits 9876");
   if (com_port == ComSerial) {
      tmatch("202\\EDIT\\1450>", "ERROR - #202 |PIN Mismatch|");
   } else {
      tmatch("  202",            "ERROR - #202 |PIN Mismatch|");
   }
   tsend(                  "unlock swlimits 6789");
   tmatch(NORMAL_PROMPT,   "unlock swlimits unlocked");
   tsend(                  "lm l0.2 dv 1000.0");
   tmatch(NORMAL_PROMPT,   "lm l0.2 dv *");
   ttimeout(EXTENDED_TIMEOUT);
   tsend(                  "restore 5");
   ttimeout(NORMAL_TIMEOUT);
   IW_sprintf(buf,         "lm l0.2 dv %s", sw_limit_buf);
   tsend(                  buf);
//
//*********************************************************
// <LS>
// Note: LS can take up more than 2 lines for a
// large number (>25) of logical units (number of modules)
//*********************************************************
   tsection("Mfcmd - LS");
   tsend(                  "ls");
   tmatch(NORMAL_PROMPT,   "ls %");
   tcopy_matched_token( measured_summary0,   1, HEX4LEN);
   tcopy_matched_token( set_summary,         2, HEX4LEN);
   if (number_of_modules > 25) {
      tmatch(NORMAL_PROMPT,   "%");
   }
   if (Lun_Map[0] == Module1461N) {
      tsend(                  "ld l0.4 dv -1000.0");
   } else {
      tsend(                  "ld l0.4 dv 1000.0");
   }
   tsend(                  "ld l0.4 ce 1");
   tinc_hex4(set_summary);
   tinc_hex4(set_summary);
   tsend(                  "ls");
#ifdef BUGFIX_0
   tmatch(NORMAL_PROMPT,   "ls * ^SET_SUMMARY^ %", set_summary);
#else
   tmatch(NORMAL_PROMPT,   "ls * * %");
#endif
   if (number_of_modules > 25) {
      tmatch(NORMAL_PROMPT,   "%");
   }
   tsend(                  "hvon");
   tdelay(5);
   tsend(                  "ls");
#ifdef BUGFIX_0
   tmatch(NORMAL_PROMPT,   "ls * ^SET_SUMMARY^ %", set_summary);
#else
   tmatch(NORMAL_PROMPT,   "ls * * %");
#endif
   tcopy_matched_token( measured_summary1,   1, HEX4LEN);
   if (measured_summary0 == measured_summary1) {
      tfail("Measured summary failed to change.");
   }
   if (number_of_modules > 25) {
      tmatch(NORMAL_PROMPT,   "%");
   }
   tsend(                  "ld l0.4 dv 0");
   tsend(                  "ld l0.4 ce 0");
   tsend(                  "hvoff");
//
//*********************************************************
//<PASSWORD> <EDIT> <VIEW>
//*********************************************************
   tsection("Mfcmd - PASSWORD");
   tsend(                   "sysdef");
   tmatch(NORMAL_PROMPT,    "sysdef 000B");
   if (com_port != ComSerial) {
      tsend(                "password 1234");
```

```
      tmatch("  220",       "ERROR - #220 |Serial Command Only|");
   } else {
      tsend(                    "password 1234");
      tmatch("164\\EDIT\\1450>", "ERROR - #164 |Passwords are disabled|");
// Turn on Maint to Enable Password
      tsend(                "maint on");
      tsend(                "sysdef 000F");
      tmatch(NORMAL_PROMPT, "sysdef 000F");
      tsend(                "maint off");
      tsend(                "password 1234");
      tmatch(NORMAL_PROMPT, "password repeat to verify");
      tsend(                "password 1234");
      tmatch(NORMAL_PROMPT, "password set");
      tsend(                    "view");
      tmatch("108\\EDIT\\1450>", "ERROR - #108 |Syntax Error|");
      tsend(                  "view 1234");
      tmatch("2\\VIEW\\1450>", "view");
      tsend(                    "ld l0.0 DV 0.");
      tmatch("176\\VIEW\\1450>", "ERROR - #176 |Session is in View Mode|");
      tsend(                  "edit 1234");
      tmatch("1\\EDIT\\1450>", "edit");
      tsend(                "view 1234");
      tmatch(NORMAL_PROMPT, "view");
      tsend(                "maint on");
      tsend(                "sysdef 000B");
      tsend(                "maint off");
      tsend(                   "edit");
      tmatch("1\\edit\\1450>", "edit");
   }
//
//*************************************************************
// <PROP>
//*************************************************************
   tsection("Mfcmd - PROP");
   tsend(                "prop l0");
   tmatch(NORMAL_PROMPT,   "prop l0 %");
   number_of_properties = tfetch_last_matched_token_count() - 2;
//
//*************************************************************
// <PS>
//*************************************************************
   tsection("Mfcmd - PS");
   tsend(                "ps l0");
   tmatch(NORMAL_PROMPT,   "ps l0 %");
   if ( (number_of_properties + 2) != tfetch_last_matched_token_count()) {
      tfail("PS returned invalid number of values");
   }
//
//*************************************************************
// <PUPSTATUS>
//*************************************************************
   tsection("Mfcmd - PUPSTATUS");
   tsend(                "pupstatus");
   tmatch(NORMAL_PROMPT,   "pupstatus 1 1 1");
//
//*************************************************************
// <SAVE> <RESTORE>
//*************************************************************
//      Save 5 save sets  (Sixth one used script default)
   tsection("Mfcmd - SAVE, RESTORE");
   tsend(                "ld l0.0 dv 0 0 0 0 0 0 0 0 0 0 0 0");
   for (i=0; i<5; i++) {
      if (Lun_Map[0] == Module1461N) {
         IW_sprintf(buf,       "ld l0.%d dv -1%d", i, i);
      } else {
         IW_sprintf(buf,       "ld l0.%d dv 1%d", i, i);
      }
      tsend(               buf);
//  Allow time for powerup save to complete
      if ( mf_fw_version < (float)2.40) {
         tdelay(15);
      } else {
         tdelay(8);
      }
      IW_sprintf(buf,       "save %d", i);
```

```
      tsend(               buf);
      IW_strcat(buf,       " Complete");
      tmatch(NORMAL_PROMPT, buf);
   }
//      Restore 6 restore sets
   ttimeout(EXTENDED_TIMEOUT);
   for (i=0; i<5; i++) {
      IW_sprintf(buf,      "restore %d", i);
      tsend(               buf);
      tmatch(NORMAL_PROMPT, "Restore In progress... Please be patient");
      IW_strcat(buf,       " Complete");
      tmatch(NORMAL_PROMPT, buf);
      tdelay(2);
      IW_sprintf(buf,      "rc l0.%d dv", i);
      tsend(               buf);
      if (Lun_Map[0] == Module1461N) {
         IW_sprintf(buf,      "rc l0.%d dv -1%d.?", i, i);
      } else {
         IW_sprintf(buf,      "rc l0.%d dv 1%d.?", i, i);
      }
      tmatch(NORMAL_PROMPT, buf);
   }
   ttimeout(NORMAL_TIMEOUT);
   tsend(                  "ld l0 dv 0 0 0 0 0 0 0 0 0 0 0 0");
//
//*************************************************************
// <SRC>
//*************************************************************
   tsection("Mfcmd - SRC");
   // Load values only into 1461N's or P's
   if (!All_1461s) {
      tmessage("Installed Modules not all 1461's");
      tmessage("Test of SRC not done.");
   } else {
      for (i=0; i<number_of_modules; i++) {
         if (Lun_Map[i] == ModuleOther) {
            tmessage(" Internal Inconsistency!! Should be all 1461's");
            texit();
         }
         k = Lun_Map[i] * (i + 1); // Use fact ModuleType is -1 for 1461N's and +1
for P's.
         IW_sprintf(buf,      "ld l%d dv %d12. %d1. %d2. %d3."
                                    " %d4. %d5. %d6. %d7."
                                    " %d8. %d9. %d10. %d11.",
                                i,     k,   k,   k,    k,
                                       k,   k,   k,    k,
                                       k,   k,   k,    k);
         tsend(               buf);
         tmatch(NORMAL_PROMPT, IGNORE_RESPONSE);
      }
      // Check values for 1461N's only - not lun = 0 must be 1 1461N!!
      tsend(                  "src dv");
      k = Lun_Map[0] ; // Use fact ModuleType is -1 for 1461N's and +1 for P's.
      IW_sprintf(buf,         "src dv %d12.0 %d1.0 %d2.0 %d3.0"
                                 " %d4.0 %d5.0 %d6.0 %d7.0"
                                 " %d8.0 %d9.0 %d10.0 %d11.0",
                                    k,    k,    k,     k,
                                    k,    k,    k,     k,
                                    k,    k,    k,     k);
      tmatch(NORMAL_PROMPT,   buf);
      for (i=1; i<number_of_modules; i++) {
         k = Lun_Map[i] * (i + 1); // Use fact ModuleType is -1 for 1461N's and +1
for P's.
         IW_sprintf(buf,               "%d12.0 %d1.0 %d2.0 %d3.0"
                                       " %d4.0 %d5.0 %d6.0 %d7.0"
                                       " %d8.0 %d9.0 %d10.0 %d11.0",
                                           k,   k,    k,     k,
                                           k,   k,    k,     k,
                                           k,   k,    k,     k);
         tmatch(NORMAL_PROMPT,         buf);
      }
      ttimeout(EXTENDED_TIMEOUT);
      tsend(                  "restore 5");
      tmatch(NORMAL_PROMPT,   "Restore in progress... Please be patient.");
      tmatch(NORMAL_PROMPT,   "restore 5 complete");
```

```
      ttimeout(NORMAL_TIMEOUT);
   }
//
//*********************************************************
// <SRM>
//*********************************************************
   tsection("Mfcmd - SRM");
   if (!All_1461s) {
      tmessage("Installed Modules not all 1461's");
      tmessage("Test of SRM not done.");
   } else {
   // Save limit to be restored later
      tsend(                  "rm l0.0 dv");
      tmatch(NORMAL_PROMPT,   "rm l0.0 dv *");
      tcopy_matched_token(sw_limit_buf,  3, 20);
   // Load values only into 1461N's
      for (i=0; i<number_of_modules; i++) {
         if (Lun_Map[i] == ModuleOther) {
            tmessage(" Internal Inconsistency!! Should be all 1461's");
            texit();
         }
         k = Lun_Map[i] * (i + 1); // Use fact ModuleType is -1 for 1461N's and +1
for P's.
         IW_sprintf(buf,      "lm l%d dv %d12. %d1. %d2. %d3."
                                    " %d4. %d5. %d6. %d7."
                                    " %d8. %d9. %d10. %d11.",
                                 i,     k,   k,   k,   k,
                                        k,   k,   k,   k,
                                        k,   k,   k,   k);
         tsend(               buf);
         tmatch(NORMAL_PROMPT, IGNORE_RESPONSE);
      }
      tsend(                  "srm dv");
      k = Lun_Map[0]; // Use fact ModuleType is -1 for 1461N's and +1 for P's.
      IW_sprintf(buf,         "srm dv %d12.0 %d1.0 %d2.0 %d3.0"
                                 " %d4.0 %d5.0 %d6.0 %d7.0"
                                 " %d8.0 %d9.0 %d10.0 %d11.0",
                                    k,    k,    k,     k,
                                    k,    k,    k,     k,
                                    k,    k,    k,     k);
      tmatch(NORMAL_PROMPT,   buf);
      for (i=1; i<number_of_modules; i++) {
         k = Lun_Map[i] * (i + 1); // Use fact ModuleType is -1 for 1461N's and +1
for P's.
         IW_sprintf(buf,               "%d12.0 %d1.0 %d2.0 %d3.0"
                                       " %d4.0 %d5.0 %d6.0 %d7.0"
                                       " %d8.0 %d9.0 %d10.0 %d11.0",
                                          k,    k,    k,     k,
                                          k,    k,    k,     k,
                                          k,    k,    k,     k);
         tmatch(NORMAL_PROMPT,         buf);
      }
   // Restore does not overwrite limits
      for (i=0; i<number_of_modules; i++) {
         IW_sprintf(buf,      "lm l%d dv %s %s %s %s %s %s %s %s %s %s %s %s",
                                 i, sw_limit_buf, sw_limit_buf, sw_limit_buf,
                                    sw_limit_buf, sw_limit_buf, sw_limit_buf,
                                    sw_limit_buf, sw_limit_buf, sw_limit_buf,
                                    sw_limit_buf, sw_limit_buf, sw_limit_buf);
         tsend(               buf);
      }
   }
//
//*********************************************************
// SYSDEF>
//*********************************************************
   tsection("Mfcmd - SYSDEF init");
   tsend(                 "sysdef");
   tmatch(NORMAL_PROMPT,  "sysdef 000B");
// Attempt to turn on password enable
   tsend(                 "sysdef 000F");
   tmatch(NORMAL_PROMPT,  "sysdef 000B");
// Attempt to turn on unassigned bits
   tsend(                 "sysdef FF0B");
   tmatch(NORMAL_PROMPT,  "sysdef 000B");
```

```
// Check Status ground on any trip and hvoff
// Setup ground on any trip
   tsend(                  "sysdef 001B");
   tmatch(NORMAL_PROMPT,   "sysdef 001B");
   tsend(                  "save 1");
// Setup to cause a channel trip low trip current and high ramp rate.
   tsection("Mfcmd - SYSDEF Status Ground on any Trip");
   if (Lun_Map[0] == Module1461N) {
      tsend(                  "ld l0.0 dv -1000.0");
      tsend(                  "ld l0.0 tc -10.0");
   } else {
      tsend(                  "ld l0.0 dv 1000.0");
      tsend(                  "ld l0.0 tc 10.0");
   }
   tsend(                  "ld l0.0 ce 1");
   tsend(                  "ld l0.0 rup 2000.0");
   tcheck_port_in(StatusPort, TTL_ON);
   tsend(                  "hvon");
   tdelay(5);
   tcheck_port_in(StatusPort, TTL_OFF);
   tsend(                  "ld l0.0 ce 0");
   tdelay(5);
   tcheck_port_in(StatusPort, TTL_ON);
// Setup ground on any trip or hvoff
   tsection("Mfcmd - SYSDEF Status Ground on trip or HVOFF.");
   tsend(                  "sysdef 003B");
   tmatch(NORMAL_PROMPT,   "sysdef 003B");
   tsend(                  "hvoff");
   tmatch(NORMAL_PROMPT,   "hvoff");
   tcheck_port_in(StatusPort, TTL_OFF);
   tsend(                  "hvon");
   tmatch(NORMAL_PROMPT,   "hvon");
   tdelay(2);
   tcheck_port_in(StatusPort, TTL_ON);
// Cause channel trip
   tsend(                  "ld l0.0 ce 1");
   tdelay(7);
   tcheck_port_in(StatusPort, TTL_OFF);
   tsend(                  "hvoff");
// Clear channel trip
   tsend(                  "ld l0.0 ce 0");
   tdelay(4);
   tcheck_port_in(StatusPort, TTL_OFF);
// clear the any trip and hvoff bits
   tsend(                  "sysdef 000B");
   tmatch(NORMAL_PROMPT,   "sysdef 000B");
   tsend(                  "hvon");
   tsend(                  "hvoff");
   tcheck_port_in(StatusPort, TTL_ON);
// Make sure options have been turned off.
   tsection("Mfcmd - SYSDEF Status options off.");
   tsend(                  "hvon");
   tsend(                  "ld l0.0 ce 1");
   tdelay(7);
   tcheck_port_in(StatusPort, TTL_ON);
   tsend(                  "ld l0.0 ce 0");
   tsend(                  "hvoff");
   ttimeout(EXTENDED_TIMEOUT);
   tsend(                  "restore 5");
   ttimeout(NORMAL_TIMEOUT);
// Setup for Restore HVON on ACFAIL
   tsection("Mfcmd - SYSDEF Restore HVON on ACFAIL enable.");
// Allow time for save
   tdelay(10);
   tsend(                  "sysdef 004B");
   tmatch(NORMAL_PROMPT,   "sysdef 004B");
   tsend(                  "hvon");
   script_accycle();
   script_init();
   tdelay(20);
   tsend(                   "hvstatus");
   tmatch(NORMAL_PROMPT,    "hvstatus hvon");
// Make sure we can turn option off
   tsection("Mfcmd - SYSDEF Restore HVON on ACFAIL disable.");
   tsend(                   "sysdef 000B");
```

```
   tmatch(NORMAL_PROMPT,    "sysdef 000B");
   script_accycle();
   script_init();
   tdelay(20);
   tsend(                   "hvstatus");
   tmatch(NORMAL_PROMPT,    "hvstatus hvoff");
// Setup for macro ground cause hvoff
   tsection("Mfcmd - SYSDEF HV off on Macro Ground.");
   tport_out(MacroPort, TTL_ON);
   tsend(                   "sysdef 008B");
   tmatch(NORMAL_PROMPT,    "sysdef 008B");
   tsend(                   "hvon");
   tmatch(NORMAL_PROMPT,    "hvon");
   tdelay(5);
   tport_out(MacroPort, TTL_OFF);
   tdelay(4);
   tsend(                   "hvstatus");
   tmatch(NORMAL_PROMPT,    "hvstatus hvoff");
// Macro high does not turn on HV
   tport_out(MacroPort, TTL_ON);
   tdelay(4);
   tsend(                   "hvstatus");
   tmatch(NORMAL_PROMPT,    "hvstatus hvoff");
// Make sure we can turn this option off
   tsend(                   "sysdef 000B");
   tmatch(NORMAL_PROMPT,    "sysdef 000B");
   tsend(                   "hvon");
   tmatch(NORMAL_PROMPT,    "hvon");
   tdelay(5);
   tport_out(MacroPort, TTL_OFF);
   tdelay(4);
   tsend(                   "hvstatus");
   tmatch(NORMAL_PROMPT,    "hvstatus hvon");
   tsend(                   "hvoff");
//
//*************************************************************
//<SYSINFO>
//*************************************************************
   tsection("Mfcmd - SYSINFO");
   tsend(                   "sysinfo");
   tmatch(NORMAL_PROMPT,    "sysinfo %");
//
//*************************************************************
// <TIME>
//*************************************************************
   tsection("Mfcmd - TIME");
   tsend(                   "time");
   tmatch(NORMAL_PROMPT,    "time *:??:??");
//
//
//*************************************************************
// 1454 - Test 3-line time out display
//*************************************************************
   if (0 == IW_stricmp("1454", mf_model)) {
      tsend(                  "maint on");
      tmatch(NORMAL_PROMPT,   "maint on");
   // Check the current display sheet number
      tsend(                  "sheet");
      tmatch(NORMAL_PROMPT,   "sheet *");
      tcopy_matched_token(buf, 1, 10);
   // If not the channel vs. value sheet (==0) then
   // probably in 3Line display, so push Enter
      if (0 != atoi(buf)) {
         IW_sprintf(buf,       "push %d", RETURN);
         tsend(                buf);
         tmatch(NORMAL_PROMPT, buf);
         tdelay(5);
         // Check that we now have sheet 0
         tsend(                   "sheet");
         tmatch(NORMAL_PROMPT,   "sheet *");
         tcopy_matched_token(buf, 1, 10);
         if (0 != atoi(buf)) {
            tsend(                "maint off");
            tfail("Unable to activate Channel vs Value Display.");
         }
```

```
      }
   // Wait for timeout to see if display changed
      tsection("Mfcmd - 3Line Timeout enabled.");
      tmessage("For the next 70 seconds,");
      tmessage(" do not touch any 1454 front panel keys.");
      tdelay(70);
      tsend(                "sheet");
      tmatch(NORMAL_PROMPT,   "sheet *");
      tcopy_matched_token(buf, 1, 10);
      if (0 == atoi(buf)) {
         tsend(             "maint off");
         tfail("Did not timeout to 3Line display.");
      }
   // Check that timeout can be disabled.
      tsection("Mfcmd - 3Line Timeout disabled.");
   // Change system default
      tsend(               "sysdef 0009");
      tmatch(NORMAL_PROMPT,"sysdef 0009");
   // Establish sheet 0
      IW_sprintf(buf,         "push %d", RETURN);
      tsend(                  buf);
      tmatch(NORMAL_PROMPT,   buf);
      tdelay(5);
      tsend(                  "sheet");
      tmatch(NORMAL_PROMPT,   "sheet *");
      tcopy_matched_token(buf, 1, 10);
      if (0 != atoi(buf)) {
         tsend(               "maint off");
         tfail("Unable to activate Channel vs Value Display.");
      }
      tmessage("For the next 70 seconds,");
      tmessage(" do not touch any 1454 front panel keys.");
      tdelay(70);
      tsend(                  "sheet");
      tmatch(NORMAL_PROMPT,   "sheet *");
      tcopy_matched_token(buf, 1, 10);
      if (0 != atoi(buf)) {
         tsend(               "maint off");
         tfail("Timeout to 3Line Display not disabled");
      }
      tsend(                "sysdef 000B");
      tmatch(NORMAL_PROMPT, "sysdef 000B");
      tsend(                  "maint off");
      tmatch(NORMAL_PROMPT, "maint off");
   }
//
//*************************************************************
// Loop repeatedly like a user
//*************************************************************
//
   script_mfloop(10);

//
//*************************************************************
// Cleanup by simply deleting all save sets, menu save files,
// and log files.
//*************************************************************

   tsection("Mfcmd - cleanup");
   tsend(                  "maint on");
   tsend(                  "del *.sav");
   tsend(                  "del *.log");
   tsend(                  "maint off");
   script_inicfg(required_fw_version, mf_model, &mf_fw_version);

// Warn tester that baud rate is not 9600, by failing script
   tsend(                  "config");
// "?" is to support other baud rates
   tmatch(NORMAL_PROMPT,   "config 0174 000? 0007 * 000B");
   tcopy_matched_token(tmp_hex4,  2, HEX4LEN);
   if (0 != IW_stricmp(tmp_hex4, "0001")) {
      tmessage("Final check of baud rate via config command.");
      tmessage("Last command in test script.");
      tmessage("Baud rate not set to factory default (9600)");
   }
```

```
   return(NORMAL);

} /* Endfunction script_mfcmd() */




static int script_inicfg(float required_fw_version, char *mf_model, float
*mf_fw_version_ptr)
{
int i, number_of_modules;
char buf[80], buffer[80], buff[20];
float mf_fw_version;

// Display and check system info
   tmessage(" ");
   tsection("Initcfg - System Information");
   tsend(                  "sysinfo");
   tmatch(NORMAL_PROMPT,   "sysinfo");
   tmatch(NORMAL_PROMPT,   "LeCroy Model: *");
   tcopy_matched_token(mf_model, 2, 4);
   tfetch_last_match_response(buf, 79);
   tmessage(buf);
   for (i=0; i<10; i++) {
      tmatch(NORMAL_PROMPT,   "%");
      if (i==3) {
        tmatch(NORMAL_PROMPT,   "FW Version: *");
        tcopy_matched_token(buf, 2, 10);
        mf_fw_version = (float) atof(&buf[1]);
      }
      tfetch_last_match_response(buf, 79);
      tmessage(buf);
   }
   if (mf_fw_version < required_fw_version) {
      IW_sprintf(buf,"Mainframe FW Version %.3f is less then required Version
%.3f",
                 mf_fw_version, required_fw_version);
      tfail(buf);
   }
   *mf_fw_version_ptr = mf_fw_version;

   if ((0 != IW_stricmp("1454", mf_model)) && (0 != IW_stricmp("1458", mf_model)))
{
      IW_sprintf(buf,"Model: %s not a 1454 or 1458", mf_model);
      tfail(buf);
   }
   tmessage(" ");

   tsection("Initcfg - Delete Powerup Save");
   tsend(                  "ll");
   tmatch(NORMAL_PROMPT,   "ll %");
   number_of_modules = tfetch_last_matched_token_count() - 1;

   tsend(                  "sysdef");
   tmatch(NORMAL_PROMPT,   "sysdef 000B");
   tsend(                  "maint on");
   tmatch(NORMAL_PROMPT,   "maint on");
// Check that backplance config, (1450-1 switches) are set correctly.
   tsend(                  "bkpcfg");
   tmatch(NORMAL_PROMPT,   "bkpcfg *");
   tcopy_matched_token(buff, 1, 19);
   if (0 != IW_stricmp(buff, "EF")) {
      if (0 == IW_stricmp(buff, "DF")) {
        tmessage("Warning: Backplane baud is 19.2K!!!");
      } else {
        IW_sprintf(buf, "Backplane configuration Got: %s  Expect: EF or DF.", buff
);
        tfail(buf);
      }
   }
   tsend(                  "del lrsmap.sav");
```

```
   tsend(                  "del *.lck");     // cleanup lock files from previous
possiblly incomplete run
   tmatch(NORMAL_PROMPT,   IGNORE_RESPONSE); // file could be already missing
   tsend(                  "maint off");
   tmatch(NORMAL_PROMPT,   "maint off");

   script_accycle();
   script_init();

   tsection("Initcfg - Establish Module Defaults");
   tsend(                  "pupstatus");
   thint_if_next_step_fails(" Possible ACFAIL interrupt problem.");
   tmatch(NORMAL_PROMPT,   "pupstatus 2 2002 1");
   tsend(                  "config");
   tmatch(NORMAL_PROMPT,   "config 0274 000? 0007 * 000B");

// Determine where 1461N's  and 1461P's are at if any
   for (i=0; i<number_of_modules; i++) {
     IW_sprintf(buf,"id l%d", i);
     tsend(                buf);
     strcat(buf,           " * * * * * * * * * *");
     tmatch(NORMAL_PROMPT, buf);
     tcopy_matched_token(buff, 2, 10);
     if (0 == IW_stricmp("1461N", buff)) {
        Lun_Map[i] = Module1461N;
     } else if (0 == IW_stricmp("1461P", buff)) {
        Lun_Map[i] = Module1461P;
     } else {
        Lun_Map[i] = ModuleOther;
        All_1461s = FALSE;
     }
     IW_sprintf(buffer, "lun: %d  %s", i, buff);
     tmessage(buffer);
   }
   /* Stop here if first module is not a 1461N or 1461P. */
   if (Lun_Map[0] == ModuleOther) {
      tmessage("First module not a 1461N or P!");
      texit();
   }
   Lun_Mapped = TRUE;
   tsend(                  "ld l0 dv 0");
   tmatch(NORMAL_PROMPT,   "ld l0 dv *");
// Give the box time to save the new values
   tdelay(10);

   script_accycle();
   script_init();

   tsection("Initcfg - Establish Normal Powerup Status and Config");
   tsend(                  "pupstatus");
   thint_if_next_step_fails(" Possible ACFAIL interrupt problem.");
   thint_if_next_step_fails(" Possibly not enough delay for save.");
   tmatch(NORMAL_PROMPT,   "pupstatus 1 1 1");
   tsend(                  "config");
// "?" is to support other baud rates
   tmatch(NORMAL_PROMPT,   "config 0174 000? 0007 * 000B");


   return(NORMAL);

} /* Endfunctio script_inicfg() */



static int script_accycle()
{
int i, status;
COM_PORTS com_port;

   tmode(ExitOnError);
   tmessage("AC Cycle Begin.");
   tcom_port(&com_port);
   if (com_port == ComSocket) {
      tclose_socket();
   }
```

```
   tport_out(ACPowerPort, PortOff);
   texpect_serial_echo(TRUE);
   tdelay(3);
   tport_out(ACPowerPort, PortOn);
   tdelay(5);

   tmode(ReturnStatus);
   tflush();

   if (com_port == ComArcnet)  {
      /* wait for recon to finish */
      while (tis_arcnet_recon_active()) {
         tmessage("ARCNET reconfig. active!");
      }
   }

   /* Here it is the power up sequence. */
   for (i=0; i<60; i++) {
      if (com_port == ComSerial) {
         tsend_nowait("");
         tdelay(1);
         tsend_nowait("");
         status = tmatch("0\\Enter \"1450\" to begin>", IGNORE_RESPONSE);
         if (status == NORMAL) {
            tsend_nowait("");
            status = tmatch("0\\Enter \"1450\" to begin>", IGNORE_RESPONSE);
            break;
         } else if (status != TSTERR_TIMEOUT) {
            break;
         }
      } else if (com_port == ComSocket) {
         status = tping(2);
         tmessage("Ping ");
         if (status == NORMAL) {
            break;
         } else if (status != TSTERR_TIMEOUT) {
            break;
         }
      } else {
         tsend_nowait("Hi");
         status = tmatch(NORMAL_PROMPT, "Hi! How are you?");
         if (status == NORMAL) {
            tflush();
            break;
         } else if (status != TSTERR_TIMEOUT) {
            break;
         }
      }
   }
   tmode(ExitOnError);
   if (status != NORMAL) {
      tstatus(status, "ACCycle Failed.");
      texit();
   } else {
      tmessage("AC Cycle Complete.");
   }
   if (com_port != ComSerial) {
      tdelay(5);
      tflush();
      tsend("Hi");
      tmatch(NORMAL_PROMPT, IGNORE_RESPONSE);
   }
   tflush();
   return(status);

} /* Endfunction script_accycle() */



static int script_init()
{
COM_PORTS com_port;
int test_with_echo_off_flag;

  tcom_port(&com_port);
```

```
  tmode(ExitOnError);
  if (com_port == ComSerial) {
     tsend_nowait("");
     tset(PropIgnorePrompt, TRUE);
     tmatch(NORMAL_PROMPT, IGNORE_RESPONSE);
     tsend("1450");
     tset(PropIgnorePrompt, FALSE);
     tmatch(NORMAL_PROMPT, IGNORE_RESPONSE);
     tsend("hi");
     thint_if_next_step_fails(" Is the front panel switch in REMOTE?");
     thint_if_next_step_fails(" Is the Panic Off clear?");
     tmatch("1\\EDIT\\1450>", "Hi! How are you?");
  } else {
     tsend("hi");
     thint_if_next_step_fails(" Is the front panel switch in REMOTE?");
     thint_if_next_step_fails(" Is the Panic Off clear?");
     tmatch("    1", "Hi! How are you?");
     tflush();
  }
  tget(PropTestWithEchoOff, &test_with_echo_off_flag);
  if (test_with_echo_off_flag) {
     tsend("echo off");
     tmatch(NORMAL_PROMPT, "echo off");
     texpect_serial_echo(FALSE);
  }
  return(NORMAL);

} /* Endfunction script_init() */



int script_test_port()
{
int on_off;

   tverify(VerifyControls);
// To handle sysinfo command and up to 16 modules
   tprep_multi_rec(20);
   tmode(ExitOnError);
   for (;;) {
      tport_in(StatusPort, &on_off);
      twait_for_any_key(ALL_TIME);
      tport_out(ACPowerPort, TTL_ON);
      tport_out(MacroPort, TTL_ON);
      twait_for_any_key(ALL_TIME);
      tport_out(ACPowerPort, TTL_OFF);
      twait_for_any_key(ALL_TIME);
      tport_out(MacroPort, TTL_OFF);
      twait_for_any_key(ALL_TIME);
      if (on_off == 40) { // Never True but makes WATCOM compiler happy
         break;
      }
   }
   return(NORMAL);
}



int script_mfloop(int number_of_times)
{
int i,j, k, number_of_modules;
VERIFY_LEVELS verify_level;
char buf[MAX_MESSAGE_LEN + 1], buffer[80], buff[20];
COM_PORTS com_port;
int hvon_flag;

   IW_sprintf(buf,"Mfloop - %d times", number_of_times);
   tsection(buf);
// Make sure remote session active
   tcom_port(&com_port);
   if (com_port == ComSerial) {
      tsend("");
      tmatch("1\\EDIT\\1450>", IGNORE_RESPONSE);
```

```
} else {
   tsend(" ");
   tmatch("    1", IGNORE_RESPONSE);
}
verify_level = tget_verify();
if (verify_level < VerifyCommands) {
   tverify(VerifyCommands);
}
tsend(                  "save 1");
tsend(                  "pupstatus");
tsend(                  "config");
tsend(                  "ll");
tmatch(NORMAL_PROMPT,   "ll %");
number_of_modules = tfetch_last_matched_token_count() - 1;

if (!Lun_Mapped) {
   // Determine where 1461N's  and 1461P's are at if any
   for (i=0; i<number_of_modules; i++) {
      IW_sprintf(buf,"id l%d", i);
      tsend(               buf);
      strcat(buf,          " * * * * * * * * * *");
      tmatch(NORMAL_PROMPT, buf);
      tcopy_matched_token(buff, 2, 10);
      if (0 == IW_stricmp("1461N", buff)) {
         Lun_Map[i] = Module1461N;
      } else if (0 == IW_stricmp("1461P", buff)) {
         Lun_Map[i] = Module1461P;
      } else {
         Lun_Map[i] = ModuleOther;
         All_1461s = FALSE;
      }
      IW_sprintf(buffer, "lun: %d  %s", i, buff);
      tmessage(buffer);
   }
   Lun_Mapped = TRUE;
}

for (i=0; i<number_of_modules; i++)  {
   if (Lun_Map[i] != ModuleOther) {
      if (Lun_Map[i] == Module1461N) {
         IW_sprintf(buf,     "ld l%d dv -1000.0 -1000.0 -1000.0 -1000.0"
                                  " -1000.0 -1000.0 -1000.0 -1000.0"
                                  " -1000.0 -1000.0 -1000.0 -1000.0", i);
      } else {
         IW_sprintf(buf,     "ld l%d dv 1000.0 1000.0 1000.0 1000.0"
                                  " 1000.0 1000.0 1000.0 1000.0"
                                  " 1000.0 1000.0 1000.0 1000.0", i);
      }
      tsend(              buf);
      IW_sprintf(buf,     "ld l%d rup 500.0 500.0 500.0 500.0"
                                 " 500.0 500.0 500.0 500.0"
                                 " 500.0 500.0 500.0 500.0", i);
      tsend(              buf);
      IW_sprintf(buf,     "ld l%d ce 1 1 1 1 1 1 1 1 1 1 1", i);
      tsend(              buf);
   }
}
for (k=0, j=0; k<number_of_times; k++, j++) {
   if (j == 0) {
      tsend(            "hvon");
      hvon_flag = TRUE;
   } else {
      hvon_flag = FALSE;
   }
   tsend(               "gs");
   for (i=0; i<number_of_modules; i++) {
      IW_sprintf(buf,   "ps l%d", i);
      tsend(            buf);
      if (Lun_Map[i] != ModuleOther) {
         IW_sprintf(buf,   "rc l%d mv", i);
         tsend(            buf);
         IW_sprintf(buf,   "rc l%d mc", i);
         tsend(            buf);
      }
      IW_sprintf(buf,   "rc l%d st", i);
```

```
         tsend(            buf);
      }
      tdelay(1);
      tsend(               "config");
      tsend(               "gs");
      if (j == 4) {
         tsend(            "hvoff");
      } else if (j == 8)  {
         j = -1;
      }
      tsend(               "src mv mc st");
   }
   tsend(                  "hvoff");
   ttimeout(EXTENDED_TIMEOUT);
   tsend(                  "restore 1");
   ttimeout(NORMAL_TIMEOUT);
   tverify(verify_level);
   return(NORMAL);
}


int script_green(int selection)
{
int i;
char buf[MAX_MESSAGE_LEN+1];
int test_with_echo_off_flag;

// <SAVE> <RESTORE>
//*********************************************************
//      Save 5 save sets  (Sixth one used script default)
  tget(PropTestWithEchoOff, &test_with_echo_off_flag);
  if (test_with_echo_off_flag) {
     tsend("echo off");
     tmatch(NORMAL_PROMPT, "echo off");
     texpect_serial_echo(FALSE);
  }
  switch(selection) {
    case GREEN_ACFAIL_CHECK:
      tsection("Mfcmd - ACFAIL check");
      script_accycle();
      script_init();
      tsend(                  "pupstatus");
      thint_if_next_step_fails(" Possible ACFAIL interrupt problem.");
      tmatch(NORMAL_PROMPT,   "pupstatus 1 1 1");
      tmessage("End cycle");
      tdelay(5);
      break;
    case GREEN_SAVE_RESTORE_CHECK:
      tsection("Mfcmd - SAVE, RESTORE");
      tsend(                  "ld l0.0 dv 0 0 0 0 0 0 0 0 0 0 0 0");
      for (i=0; i<5; i++) {
         if (Lun_Map[0] == Module1461N) {
            IW_sprintf(buf,       "ld l0.%d dv -1%d", i, i);
         } else {
            IW_sprintf(buf,       "ld l0.%d dv 1%d", i, i);
         }
         tsend(                buf);
//  Allow time for powerup save to complete
#ifdef BUGFIX_1
#else
//         tdelay(10);
#endif
         IW_sprintf(buf,       "save %d", i);
         tsend(                buf);
         IW_strcat(buf,        " Complete");
         tmatch(NORMAL_PROMPT, buf);
      }
      //        Restore 6 restore sets
      ttimeout(EXTENDED_TIMEOUT);
      for (i=0; i<5; i++) {
         IW_sprintf(buf,       "restore %d", i);
         tsend(                buf);
         tmatch(NORMAL_PROMPT, "Restore In progress... Please be patient");
         IW_strcat(buf,        " Complete");
         tmatch(NORMAL_PROMPT, buf);
```

```
         IW_sprintf(buf,       "rc l0.%d dv", i);
         tsend(                buf);
         if (Lun_Map[0] == Module1461N) {
            IW_sprintf(buf,       "rc l0.%d dv -1%d.?", i, i);
         } else {
            IW_sprintf(buf,       "rc l0.%d dv 1%d.?", i, i);
         }
         tmatch(NORMAL_PROMPT, buf);
      }
      ttimeout(NORMAL_TIMEOUT);
      tsend(                   "ld l0 dv 0 0 0 0 0 0 0 0 0 0 0 0");
//
//*********************************************************
      tmessage("Script Green Complete");
      break;
   }

   return(NORMAL);
}
```

Test Functions

In this section, all possible test functions are listed. Eventually, descriptions of the operation of these functions will also be included. Meanwhile, it is hoped that the function names and the previous test script listing is sufficient documentation of these functions. The test functions tsend(), tmatch(), and tverify() were described in detail in a previous section.

```
int tac_power_mode(TEST_MODES mode)

int tcheck_port_in(TTL_PORTS port, int ttl_on_off)

int tcom_port(COM_PORTS *com_port_ptr)

int tcopy_matched_token(char *buffer, int token_number, unsigned int buffer_len)

void tdelay(int seconds)

void terror(ERROR_LEVELS level, int status, char *buffer)

void texit()

void texpect_serial_echo(int flag)

void thint(char *buffer)

void tfail(char *buffer)

TEST_MODES tfetch_ac_power_mode(void)

int tfetch_last_error_flag(void)

char * tfetch_last_prompt(void)

int tfetch_last_match_response(char *buffer, int len)

int tfetch_last_matched_token_count(void)

void tflush(void)

VERIFY_LEVELS tget_verify(void)

int tget(PROPERTIES prop, int *value_ptr)

int thint_if_next_step_fails( char *buffer)

int tinc_hex4(char * str)

int tinit(OP_MODES op_mode, int max_buffer_len, ENTRY_STRUCT *p_ptr, int
timeout_seconds)

void tmessage(char *buffer)
```

```
int tlogtofile(char *filename)
int tmode(OP_MODES op_mode)
unsigned int tport_base_address()
int tport_init(PORT_ADDRESS_NUMBERS number)
int tport_in(TTL_PORTS port, int *on_off_ptr)
int tport_out(TTL_PORTS port, int on_off)
int tprep_multi_rec(int no_responses)
int tmatch(char *prompt_status, char *response, ...)
int tsection(char *buffer)
int tsend(char* buffer)
int tsend_nowait(char* buffer)
int tset(PROPERTIES prop, int value)
int tstatus(int status, char* buffer)
void tstepit(int on_off)
void ttimeout( int timeout_seconds)
void tverify(VERIFY_LEVELS level)
void tversion(char *buffer)
int twait_for_any_key( int wait_in_seconds)
```

int tyes_or_no(int* yes_or_no_ptr, int wait_in_seconds, int def)

24V Switching Power Supply Over Load Test

When the Main Frame 24V power supply is over loaded a 24V fail error will occur. Load the Main frame with the number of active switch as shown in Table 1 using Load Card P/N 1589-58 for the 1458xx or 1589-54 for the 1454 which will cause an over load condition on the 24V. Verify that the Main Frame reports a 24V FAIL condition. The failure condition is when the unit does not report a 24V FAIL error.

Burn-in

Burn-in Recommendations

Each 1458 Main Frame shall be tested in the environmental chamber and go through four consecutive burn-in cycles as stated below with out failure. The 1458/54 is to be loaded with one 1461X card plus load cards as shown in Table 1

Environmental Conditions

Total Burn-In Cycle: 24 hours (as follows:)

- 1 hours @ 45 °C
- 2 hours @ 5 °C
- 3 hours @ 45 °C

Typical Slew Time Between Temperatures is 5 minutes

LOAD CARD CONFIGURATION				
Main Frame Version:	1458LP	1458	1458HP	1454
Number of Load cards P/N 1589-58:1458xx, 1589-54:1454	13	13	13	3
Number of Active 24V Loads	4	8	12	2
Number of Active 24V Loads for Over Load	5	9	13	3

Table 1

Criteria for successful completion of burn-in is that the HV remain ON at the end of the cycle and that the ops log does not report any reboots during the burn-in cycle.

Re-Burn-in Recommendations

If the Mainframe fails the 1st burn-in the unit must go through two additional burn-in cycles with out failure. If during one of these cycles the unit fails again then the mainframe must be removed from the chamber and a thorough test must be done to find the fault.

Pre-Shipment Cleanup

Overview

Once the system testing is complete, a number of operations need to be done to prepare the unit for shipment. These include deleting files on the "D:" drive, checking that system defaults are set to factory default values, and assuring a clean powerup for the shipped module configuration.

Procedure

Install any modules that will be shipped with or in the mainframe.

Configure for **VT100 Testing**.

1. **Check date and time correct.**
   A. At the serial prompt, check the time> "TIME".
   B. Check the date> "DATE".
2. **Clear op hours**
   A. After command prompt enter Maintenance mode> " MAINT ON".
   B. Delete hour.ops file on D: drive> "DEL hours.ops".
   C. Using the directory command > "DIR", check that only files on D: drive are:
      1. 1454 - The file holding the screen contrast - contrst.cfg.
      2. The file holding the current slot map - lrsmap.sav (Possibly not there).
      3. The powerup save files containing demand values - lrs#.sav (Possibly not there).
      4. The powerup save files containing limit values - lrsl#.sav (Possibly not there)..
   D. Exit Maintenance mode> "MAINT OFF".
3. **Check Backplane Baud Rate**
   A. Via System menu, check System information is correct and that op hours is zero.
   B. Select "0) Powerup Status".
   C. Observe next to last line of display is "Backplane baud: 38400".
4. The following steps are now done in the "Cleanup" section of the Test script thread and listed here for completeness.
   A. Clean D: drive
      1. After command prompt enter Maintenance mode> " MAINT ON".
      2. Delete log files on D: drive> "DEL *.log".
      3. Delete all save files on D: drive> "DEL *.sav".
      4. Check that only file on D: drive is "contrst.cfg" > "DIR".
      5. Exit Maintenance mode> "MAINT OFF".
      6. Power down mainframe.
   B. Re-establish module defaults as mainframe save sets (after cleaning D: drive)
      1. Powerup Mainframe.
      2. Observe 1 beep followed later (after remote/network flashing ends) by 2 beeps.
      3. 1454 - Observer screen message that no save sets were found.
      4. 1458 - establish VT100 session.
      5. Move the cursor over a demand voltage and push *0* then *ENTER*.

6. Notice slight delay in response since save sets are being written to internal battery backed BRAM.
7. Power mainframe down then up.
8. Observe 1 beep followed later by 1 beep (Normal powerup with valid save sets).

C. Check System Information, Backplane baud , and EEPROM write disabled.

1. From the System menu, SELECT “System Information”.
2. Escape to System Information menu.
3. Position cursor over “Test Date” and push *ENTER*.
4. If value entry display appears EEPROM write is enabled on the 1450-1. (Enter proper Test Date and hit return.) THIS MUST BE DISABLED BEFORE SHIPMENT.
5. If nothing happens after pushing *ENTER*, then EEPROM write is properly disabled.

D. Check that External Serial and ARCNET configured for factory default.

1. From the system menu, SELECT “Serial Setup”.
2. Check that Serial Setup is Baud: 9600, Data Bits: 8, Stop Bits 1, Parity: None, Echo: On
3. 1458 - Baud rate determined by 1450-1 jumper settings, reset of serial setup “hardwired”.
4. 1454 - Used *ARROW* and *SELECT* keys to make changes if needed, use *SAVE* to store changes.
5. Push ESC, then select “Network Setup”.
6. Check that ARCNET Node ID is 7.
7. ARCNET Node ID controlled by card switches (on external panel).
8. Cycle mainframe power to make any change in ARCNET Node ID take effect.

E. Check that System Defaults are configured for factory default.

1. From the system menu, SELECT “System Defaults”
2. Check that the following system default options are set YES:
   a) Timeout to 3Line Display.
   b) STATUS gnd on panic off.
3. Check that all other options are set NO.

Test Checklist

Included below is a convenient checklist of test procedure headings, which is provided as an aid to the test technician. As indicated in the first section of this document, headings which are bolded and underlined indicate tests which need to be done for each mainframe. Headings which are simply bolded need only be done one mainframe per manufacturing lot. Other headings indicate tests which should be done upon a mainframe firmware release.

Mainframe Initiation

1. **Check for correct CMOS configuration**
2. **Check HW configuration**
3. **Install Ethernet Option if desired**
4. **HV mainframe low level Tests**
5. **To fix EEPROM failure, zap the Mainframe EEPROM**
6. **Check operation of 24V Bad.**
7. **1458 - Set Housekeeping 21V**
8. **Confirm Panic Switch and Interlock Hardware work.**
9. **Affix Serial Number sticker to Mainframe and note number.**
10. **Enter Mainframe System Information.**
11. **Confirm Mainframe System Information.**
12. **Error Beep Sequence and watchdog reset checks (requires keyboard) (just step 11-A).**
13. Mainframe Hardware problem Beep Sequence (requires top or front cover off).
14. **Restore HVON after ACFail Test 5 - Disabled by system error (requires keyboard).**
15. **Check that the ARCNET card jumper JP-5 has been removed.**
16. **Check that backplane connectors have all nuts and bolts removed or that connector nuts are on solder side of backplane.**
17. **1458 - Tiewrap internal cables and wires to prevent fan interference.**
18. **1458 - Check clearance between front panel and power support tray.**
19. **Tiewrap 1450-1 to motherboard.**
20. **RTV Motherboard power connector, speaker and reset wires.**
21. **Check that notherboard ground is connected.**
22. **Mainframe Initiation closeup**

Local Interface Tests

Safety Features

1. **Panic Off Test 2 - Panic Off cannot be reset in REMOTE, only in LOCAL.**
2. **Panic Off Test 3 - Panic Off turns off HV**
3. **Panic Off Test 4 - Panic Off via Interlock**
4. **Panic Off Test 1 - Panic Off prevents HVON and other changes**
5. **Panic Off Test 5 - Panic Off remembered after Power Down**
6. **Key Switch Remote Test**
7. **Key Switch Local Test**
8. Restore HVON after ACFail, Test 1 - Exercise the option. (Exercised by hvtest.exe)
9. **Restore HVON after ACFail, Test 2 - Inhibited with front panel Standby.**
10. **Restore HVON after ACFail, Test 3 - Abort HVON countdown.**
11. **Restore HVON after ACFAIL, Test 4 - Disabled upon new module configuration.**
12. **Restore HVON after ACFAIL, Test 5 - Disabled by Panic Off.**
13. **Leave mainframe configured for "HVON ACFAIL, Restore HVON No".**

Powerup Beeps

1. **Normal Beep sequence**
2. **Warning Beep Sequence - new module configuration**
3. **Warning Beep Sequence - no module configuration found.**
4. **No Modules Installed Beep Sequence**

Manual Key Check and Value Entry

1. Enter a value.
2. Escape from value entry.
3. **Enter a range of values.**
4. Enter a module of values.
5. Delta a value (increase).
6. **Delta a module of values (decrease)**
7. Increment a value.
8. Move place value of increment then increment.
9. Decrement a value.
10. Save current values.
11. Restore values.
12. Enable a channel
13. Disable a channel
14. Turn on HV
15. Turn off HV
16. Next and Previous display sheet.
17. Hardcopy , Group, Display Up, Display Dn disabled.

18. System Menu

19. 1454 - Chan button.

20. HV ERROR LED's

21. 1454 - Module channel tripped jump.

22. 1454 - Module Jump buttons

23. 1454 - Module channel ramping jump.

24. Remove powerup save set.

25. Display Up.

26. Help.

System Menu Driven Features

1. No Action Menu escape timeout.
2. *0) Activate Three Line Display*
3. *1) Lock/Unlock Settings/Limits*
4. *2) Software Limits*
5. *3) System Defaults*
6. *4) Serial Port Setup* - FW feature
7. ***1454 - 4) Serial Port Setup - try all baud rates***
8. ***1458 - 4) Serial Port Setup - try all baud rates***
9. *7) System Information*
10. ***8) 1454 - Screen Contrast***
11. *9) Reset PANIC OFF*

VT100 Monitoring - Access Modes

1. Test Local mode.
2. Test VIEW command.
3. Test VT100 VIEW command.
4. Test Panic Off

Remote Interface Testing

1. **ARCNET testing with HVTEST.**
2. **Ethernet (1450-ET) Testing with BSDTEST.**
3. **RS-232 testing with HVTEST.**
4. **ARCNET testing - no modules.**
5. **RS-232 testing - no modules.**

Pre-Shipment Cleanup

1. **Check date and time correct.**
2. **Clear op hours**
3. **Check Backplane Baud Rate**