COS DCE BOOT FSW v1.09 Component Test Results
Requirement 5.2.3.1 Housekeeping Response Within One Second

Date: February 13, 2001
Document Number: COS-03-0025
Revision: Initial Release
Contract No.: NAS5-98043
CDRL No.: N/A

Prepared By: Tim Swanson, Software Test Engineer, Design_Net Eng.
Reviewed By: K. Brownsberger, COS Sr. Software Scientist, CU/CASA
Reviewed By: Grant Blue, COS Software & Operations Manager, BATC
Approved By: Barry Welsh, FUV Detector Program Manager, UCB
Approved By: John Andrews, COS Experiment Manager, CU/CASA

Center for Astrophysics & Space Astronomy
University of Colorado
Campus Box 593
Boulder, Colorado 80309
<table>
<thead>
<tr>
<th>Letter</th>
<th>ECO No.</th>
<th>Description</th>
<th>Check</th>
<th>Approved</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td></td>
<td>Initial Release</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Original Release</th>
<th>Name</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drawn:</td>
<td>K. Brownsberger</td>
<td>2-13-01</td>
</tr>
<tr>
<td>Reviewed:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Approved:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

THE UNIVERSITY OF COLORADO  
At Boulder  
The Center for Astrophysics and Space Astronomy  

COS DCE BOOT FSW v1.09 Component Test Results  
Requirement 5.2.3.1 Housekeeping Response Within One Second  

<table>
<thead>
<tr>
<th>Size</th>
<th>Code Indent No.</th>
<th>Document No.</th>
<th>Rev</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>COS-03-0025</td>
<td>-</td>
</tr>
</tbody>
</table>

Scale: N/A
Table of Contents

1. Introduction ............................................................................................................... 2
   1.1 Purpose .................................................................................................................. 2
   1.2 Scope ..................................................................................................................... 2
   1.3 Limitations and Constraints .................................................................................. 2
   1.4 Procedure Overview ............................................................................................. 2
   1.5 Theory of Test ....................................................................................................... 3
   1.6 Test Script Implementation .................................................................................. 3
       1.6.1 Explanation of the CHECK Directive .......................................................... 3
       1.6.2 Test Script Arguments .................................................................................. 5
2. Special Instructions ...................................................................................................... 5
   2.1 Quality Assurance ................................................................................................. 5
   2.2 Safety .................................................................................................................... 5
       2.2.1 Personal Safety .............................................................................................. 5
       2.2.2 Test Article and Equipment Safety ............................................................... 5
   2.3 Contamination ....................................................................................................... 6
3. Support Requirements ................................................................................................. 6
   3.1 Personnel .............................................................................................................. 6
   3.2 Tools, Equipment, and Materials ......................................................................... 6
   3.3 Data/Software ....................................................................................................... 7
   3.4 Required Documentation ..................................................................................... 7
4. Procedure/Task Steps ................................................................................................. 7
   4.1 Pre-Operation Activities ..................................................................................... 7
       4.1.1 Make Sure that hks Tools Are Active ......................................................... 7
       4.1.2 Make Sure that the Proper ROM Is Installed ............................................. 7
       4.1.3 Log In to the EGSE ..................................................................................... 7
       4.1.4 Set Current Directory .................................................................................. 8
       4.1.5 slogin as eagcos ....................................................................................... 8
       4.1.6 Set Current Directory .................................................................................. 8
       4.1.7 Ensure that Proper Files are Present ......................................................... 9
   4.2 Operation Execution ............................................................................................. 9
       4.2.1 Establish Initial Test Conditions ................................................................. 9
       4.2.2 Execute the Script ....................................................................................... 9
   4.3 Post-Operation Activities ................................................................................... 12
       4.3.1 Copy Reports to PC Files and Print Them ................................................. 12
       4.3.2 Complete The Test Procedure Form ....................................................... 12
1. **INTRODUCTION**

1.1 **PURPOSE**

This document presents the Cosmic Origins Spectrograph (COS) Device Control Electronics (DCE) Flight Software (FSW) certification procedure. The purpose of this procedure is to verify that the FSW satisfies Software Requirements according to the method specified in the DCE FSW Test Plan (STP).

1.2 **SCOPE**

This test procedure comprises the steps necessary to verify that the FSW satisfies Software Requirements Document (SRD) paragraph 5.2.3.1 — Housekeeping Response within One Second.

1.3 **LIMITATIONS AND CONSTRAINTS**

This test cannot be run in parallel with any other commanding activity directed at the DCE FSW (such as, for example, the periodic transmission of NOOP commands). Test hardware shall be visually inspected, and its configuration noted, prior to conducting this test.

1.4 **PROCEDURE OVERVIEW**

The procedure requires the hks tools running on the Sun SparcStation Electronic Ground Support Equipment (EGSE) whose network IP address is one of:

- `shorty.ssl.berkeley.edu`
- `taiyo.ssl.berkeley.edu`
- `ginger.ssl.berkeley.edu`

Test time shall be scheduled in advance. The Test Conductor must be logged into the Unix system as user `eageos`, and be commanding from the appropriate directory. This directory contains both the test script file and the shell script file; these two files control test execution. The test is conducted by invoking the shell script. This shell script in turn invokes the Perl 5 program `UniScript.pl`, which resides in its own distinct directory. The test procedure steps have been pre-recorded in the test script file, and are executed interpretively by the `UniScript` program. The shell script and test script are attached to this document as appendices. As `UniScript` executes the test script it sends results to the operator console and to two report files, which are also placed in the current directory. After completion of the test script, the Test Conductor can certify successful test
execution by examining the contents of the report files and determining that required outputs are present in them. Printed copies of the report files are attached to the manually completed checklist (Paragraph 4 below) as documentation of the test.

1.5 THEORY OF TEST

The essence of the test is to keep track of the ongoing “time-line” as commands are sent to, and housekeeping data returned from, the DCE. Since the Unix operating system does not provide timer services with less than 1-second resolution, the script makes use of the elapsed time values maintained by the FSW: specifically, in addition to the LFCTIME variable provided in the HK data, another variable, used internally by FSW, namely mTICKS, provides .02-second granularity. Hence, the epoch since the last timer reset (power-on or watchdog), can be determined to within a fiftieth of a second as LFCTIME + mTICKS/50. It is verified that the HK data following a command does not lag the previous data by more than .9 seconds by remembering the “preceding” time, comparing it with the “current time”, and, if the difference is less than .9 seconds, making the “current time” the new value of “preceding time”, sending the next command, and so on. An initial LFDNOOP command is sent to provide an “origin” for the time values.

1.6 TEST SCRIPT IMPLEMENTATION

1.6.1 Explanation of the CHECK Directive

For the purposes of this section, the term “system time” is defined to mean the sum of the HK variable LFCTIME and 1/50th of the value of the FSW variable mTICKS (at 0x2460 in the Patchable Constants). The latter variable counts 20-ms “ticks” since the last incrementation of LFCTIME, but is not automatically made available in the HK data. The script acquires its value by setting memory monitor 7 to 0x2460, then using the (8-bit) value LFDMONS[7] in the HK data. The script also uses two Perl scalar variables, $xt and $yt, representing, respectively, system time computed from the previous HK packet, and system time computed from the current packet. $xt is initialized by means of a CHECK directive that always succeeds, namely

CHECK 1,((xt=LFCTIME+LFDMONS[7]/50.0)==$xt)

These variables are “automatically re-vivified” (see Perl documentation) for this script by virtue of their occurrence in the Perl-expression argument of a UniScript CHECK directive; they are not “standard” Perl variables like $B1, $CRC1, etc.
Verifying the requirement that each HK pack arrive no later than .9 seconds after the preceding one (assuming commanding at .9-second intervals) is equivalent to executing the following simple segment of Perl code after reception of each HK packet:

```perl
$yt = $LFCTIME+$LFDMONS[7]/50.0; # seconds + (fiftieths of a second)/50
if ($yt <= $xt + .9)      # should not be later than previous system time + .9sec
    {$xt = $yt}            # if OK, update $xt to current system time
else
    {$xt = 0}              # if not, set $xt=0; CHECK will discover this
if ($xt == 0)
    {terminate the script}
```

It is possible, owing to the ingenious quiddities of Perl assignment (=) and conditional-value (?:) operators, to compress this segment into a single Perl expression, and hence to incorporate it into a single UniScript CHECK directive. This is done as follows. The Perl expression

```perl
($yt=$LFCTIME+$LFDMONS[7]/50.0) <= $xt+.9
```

compares the current system time with that of the previous HK packet; but it also has the “side-effect” of assigning the current system time to $yt (as a floating-point number). Furthermore, the value of the expression is either true or false, and so may be used as the 1st operand of a Perl “conditional operator” ?:. Hence the expression

```perl
(($yt=$LFCTIME+$LFDMONS[7]/50.0) <= $xt+.9) ? $yt : 0
```

evaluates to $yt (if $yt is no later than $xt + .9 seconds in the system time epoch) — or to 0 (if the current HK packet arrived too late to satisfy the software requirement 5.2.3.1). The expression

```perl
$xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0) <= $xt+.9) ? $yt : 0)
```

therefore assigns to $xt either the current system time, $yt, or 0, depending, in effect, on whether the test requirement was verified or not. However, this expression, in addition to assigning a value to $xt, also itself takes on the assigned value; hence its value (namely $xt) may be compared with 0, the “error value”. The result of this greater-than (>) comparison, either true (i.e., test succeeded) or false (test failed), is the condition checked by the script statement

```perl
CHECK 1,((($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=($xt+.9)?$yt:0)>0)
```

\[x1 \neq x2 \Rightarrow x3\] takes the value \(x2\) if \(x1\) is true, otherwise the value \(x3\).
1.6.2 Test Script Arguments

The script is parameterized as shown in the following Table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
<th>Correct Argument for Version 1.09</th>
</tr>
</thead>
<tbody>
<tr>
<td>#0</td>
<td>Absolute hex storage address of intermediate “scratch” buffer for ROM data</td>
<td>C000</td>
</tr>
<tr>
<td>#1</td>
<td>Absolute hex storage address, + 256, of intermediate “scratch” buffer for ROM data</td>
<td>C100</td>
</tr>
<tr>
<td>#2</td>
<td>Absolute hex storage address of FSW NO_OPER subroutine</td>
<td>0330</td>
</tr>
<tr>
<td>#3</td>
<td>Absolute hex storage address of FSW mTICKS byte (Patchable Constants)</td>
<td>2460</td>
</tr>
</tbody>
</table>

These parameters must be encoded into the shell script u (see Appendix A).

2. SPECIAL INSTRUCTIONS

2.1 QUALITY ASSURANCE

QA support is required to verify the configuration and setup environment as well as monitoring test steps and verifying results.

2.2 SAFETY

2.2.1 Personal Safety

To ensure the safety of the test personnel during test execution the guidelines contained in Paragraph 3.4, Reference [1] will be adhered to.

2.2.2 Test Article and Equipment Safety

To ensure the safety and well-being of the COS operations bench, SITS, and related test equipment, the following primary safety requirements will be in effect during the execution of this test procedure:

- If access within one (1) meter of COS bench electronics is necessary, wrist straps attached to technical ground shall be used by all personnel involved in handling of
any COS test article. Overcurrent and overvoltage shall be set to remove power if nominal limits are exceeded.

- **Emergency Power Shutdown** — If, during the COS DCE FSW test, power is ON and a severe test equipment failure results in the power system exceeding specified limits, the Test Conductor shall direct or perform shutdown of power.

2.3 CONTAMINATION

All flight hardware shall be handled with clean latex gloves; it shall be covered with clean ESD material and/or stored in a clean flow-bench.

3. SUPPORT REQUIREMENTS

3.1 PERSONNEL

Execution of the COS DCE FSW certification procedure requires the following personnel (to be completed at the Test Readiness Review (TRR):

- **Test Director:** ____________________________________________
- **Test Conductor:** ____________________________________________
- **Test Technician:** ____________________________________________
- **QA:** ____________________________________________

3.2 TOOLS, EQUIPMENT, AND MATERIALS

The following is a list of tools, equipment, or materials required in this test. Record manufacturer and model, metrology, or property numbers of equipment used, where appropriate. Record calibration due dates where appropriate.

- **Boot Mode ROM:** schematic **27C256**

- **Engineering Ground Support Equipment** (see paragraph [1.4]. Indicate specific configuration:

<table>
<thead>
<tr>
<th>EGSE</th>
<th>DCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>taiyo</td>
<td>shorty</td>
</tr>
<tr>
<td>ginger</td>
<td>ETU</td>
</tr>
<tr>
<td></td>
<td>DCE #1</td>
</tr>
<tr>
<td></td>
<td>DCE #2</td>
</tr>
</tbody>
</table>
3.3 **DATA/SOFTWARE**

The following files must be present:

<table>
<thead>
<tr>
<th>Table 3-1: Required Program and Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>EGSE (shorty) Directory</td>
</tr>
<tr>
<td>\disks\galex\users\galex\tcs\uniscript\</td>
</tr>
<tr>
<td>\disks\galex\users\galex\tcs\uniscript\stp5_2_3_1\</td>
</tr>
<tr>
<td>Ditto</td>
</tr>
</tbody>
</table>

In addition, the hks tools must be active. Directions for activating hks are given in UCB-COS-DOC-1118 (Paragraph 3.4, Reference [4]).

3.4 **REQUIRED DOCUMENTATION**

<table>
<thead>
<tr>
<th>Reference</th>
<th>Document Number</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NHB 1700.1(V1-A)</td>
<td>NASA Basic Safety Manual</td>
</tr>
<tr>
<td>2</td>
<td>COS-03-0025</td>
<td>DCE FSW Test Procedure 5.2.3.1 (this document)</td>
</tr>
<tr>
<td>3</td>
<td>UCB-COS-008</td>
<td>COS FUV Detector Software Test Plan</td>
</tr>
<tr>
<td>4</td>
<td>UCB-COS-DOC-1118</td>
<td>COS EGSE Startup Procedure</td>
</tr>
</tbody>
</table>

4. **PROCEDURE/TASK STEPS**

4.1 **PRE-OPERATION ACTIVITIES**

4.1.1 Make Sure that hks Tools Are Active

Follow the procedure given in Paragraph 3.4, Reference [4].

4.1.2 Make Sure that the Proper ROM Is Installed

Visually verify that the ROM under test is installed: if EEPROM, in U18: if PROM, in U2 and U7.

4.1.3 Log In to the EGSE
In the following steps, the EGSE system ("taiyo") may be any of the systems listed in Paragraph 1.4 Output, from either the Unix system or from UniScript, to the Telnet terminal is represented in the typeface. Input from the Test Conductor is represented in the -Bold typeface.

<table>
<thead>
<tr>
<th>Step</th>
<th>Operator Entry/System Response</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C:\tcs\us&gt; telnet taiyo.ssl.berkely.edu</td>
<td>Establish connection to taiyo via Telnet client program</td>
</tr>
</tbody>
</table>
| 2    | Login: tcs  
     | Password: | Using telnet window, login as user tcs |

4.1.4 Set Current Directory

<table>
<thead>
<tr>
<th>Step</th>
<th>Operator Entry/System Response</th>
<th>Description</th>
</tr>
</thead>
</table>
| 3    | tcs@taiyo% cd ~galex/tcs  
     | tcs@taiyo% pwd  
     | /disks/galex/users/galex/tcs | Change current directory as shown |

4.1.5 slogin as eagcos

<table>
<thead>
<tr>
<th>Step</th>
<th>Operator Entry/System Response</th>
<th>Description</th>
</tr>
</thead>
</table>
| 4    | tcs@taiyo% slogin –l eagcos taiyo.ssl.berkeley.edu  
     | eagcos@taiyo.ssl.berkeley.edu’s password: (get from SSL personnel)  
     | Last login: Sat Oct 7 10:41:05 2000 from auntem.ssl.berkeley.edu  
     | Sun Microsystems Inc. SunOS 5.8 Generic February 2000  
     | You have mail.  
     | COS EGSE software version: devel | slogin as eagcos; get password from SSL personnel |

4.1.6 Set Current Directory

<table>
<thead>
<tr>
<th>Step</th>
<th>Operator Entry/System Response</th>
<th>Description</th>
</tr>
</thead>
</table>
| 5    | eagcos:taiyo% cd  
     | /disks/galex/users/galex/tcs/uniscript/stp5_2_3_1  
     | eagcos:taiyo% pwd  
     | /disks/galex/users/galex/tcs/uniscript/stp5_2_3_1 | Change current directory as shown |
4.1.7 Ensure that Proper Files are Present

<table>
<thead>
<tr>
<th>Step</th>
<th>Operator Entry/System Response</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>eagcos@taiyo% <strong>ls -l</strong></td>
<td>List files; the .tst file and the shell script u should be present</td>
</tr>
<tr>
<td></td>
<td>Total 12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-rw-r--r-- 1 tcs eag 1398 Oct 8 18:03 stp5_2_3_1a.tst</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-rw-r--r-- 1 tcs eag 62 Oct 9 17:44 u</td>
<td></td>
</tr>
</tbody>
</table>

4.2 OPERATION EXECUTION

4.2.1 Establish Initial Test Conditions

<table>
<thead>
<tr>
<th>Step</th>
<th>Operator Entry/System Response</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>eagcos:taiyo% <strong>set path=(~dbb/scripts/bin)</strong></td>
<td>Set path as shown to enable access to hks tools</td>
</tr>
</tbody>
</table>

4.2.2 Execute the Script

<table>
<thead>
<tr>
<th>Step</th>
<th>Operator Entry/System Response</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>eagcos:shorty% <strong>sh u</strong></td>
<td>Shell to u. You should see the accompanying output as UniScript executes</td>
</tr>
<tr>
<td></td>
<td>$pstring=C000,C100,0330,2460,0,0,0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Parameters are: Script File: stp5_2_3_1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#0: C000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#1: C100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#2: 0330</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#3: 2460</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#4: 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#5: 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#6: 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#7: 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Report file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt;/disks/galex/users/galex/tcs/uniscript/stp5_2_3_1/stp5_2_3_1.rp1 successfully opened. Report file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt;/disks/galex/users/galex/tcs/uniscript/stp5_2_3_1/stp5_2_3_1.rp2</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>Operator Entry/System Response</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>successfully opened.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Script file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/disks/galex/users/galex/tcs/uniscript/stp5_2_3_1/stp5_2_3_1.tst</td>
<td>successfully opened at level 0.</td>
</tr>
<tr>
<td></td>
<td>&quot;Press Y when ready to conduct test 5.2.3.1&quot;</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>Continuing.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Sending POR, collecting initial HK&quot;</td>
<td>LFDMAADDR 7,mTICKS,EXTERN</td>
</tr>
<tr>
<td></td>
<td>WAIT 0: HKV0=1; HKV1=66; wc=5</td>
<td>&quot;Sending LFDNOOP to get command stream started&quot;</td>
</tr>
<tr>
<td></td>
<td>LFDNOOP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WAIT 0: HKV0=2; HKV1=1; wc=0</td>
<td>&quot;Sending LFDCOPY&quot;</td>
</tr>
<tr>
<td></td>
<td>LFDCOPY  SOURCE,SOURCE,NBYTES,BANK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WAIT 0: HKV0=3; HKV1=2; wc=0</td>
<td>&quot;Sending LFMCRC&quot;</td>
</tr>
<tr>
<td></td>
<td>LFDCRC  SOURCE,NBYTES,CODE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WAIT 0: HKV0=4; HKV1=3; wc=0</td>
<td>&quot;Sending LFDDIAGC&quot;</td>
</tr>
<tr>
<td></td>
<td>LFDDIAGC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WAIT 0: HKV0=5; HKV1=4; wc=0</td>
<td>&quot;Sending LFDDNLOD&quot;</td>
</tr>
<tr>
<td></td>
<td>LFDDNLOD  SOURCE,NBYTES</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>Operator Entry/System Response</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>WAIT 0: HKV0=6; HKV1=4; wc=0</td>
<td>&quot;Sending LFDGOTO&quot;</td>
<td></td>
</tr>
<tr>
<td>LFDGOTO</td>
<td>NOOP</td>
<td></td>
</tr>
<tr>
<td>WAIT 0: HKV0=7; HKV1=6; wc=0</td>
<td>&quot;Sending LFDHKREQ&quot;</td>
<td></td>
</tr>
<tr>
<td>LFDHKREQ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAIT 0: HKV0=8; HKV1=7; wc=0</td>
<td>&quot;Sending LFDMADDR&quot;</td>
<td></td>
</tr>
<tr>
<td>LFDMADDR 0,SOURCE,DATA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAIT 0: HKV0=9; HKV1=8; wc=0</td>
<td>&quot;Sending LFDUPLOD&quot;</td>
<td></td>
</tr>
<tr>
<td>LFDUPLOD DEST,NBYTES,0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAIT 0: HKV0=10; HKV1=9; wc=0</td>
<td>&quot;Sending LFDWDOG&quot;</td>
<td></td>
</tr>
<tr>
<td>LFDWDOG 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAIT 0: HKV0=11; HKV1=10; wc=0</td>
<td>&quot;Sending LFDNOOP&quot;</td>
<td></td>
</tr>
<tr>
<td>LFDNOOP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAIT 0: HKV0=12; HKV1=11; wc=0</td>
<td>&quot;Test stp5.2.3.1 completed successfully&quot;</td>
<td></td>
</tr>
</tbody>
</table>
4.3 POST-OPERATION ACTIVITIES

4.3.1 Copy Reports to PC Files and Print Them

Using an FTP client, copy the u, stp5_2_3_1.tst, stp5_2_3_1.rp1, and stp5_2_3_1.rp2 files to appropriate PC files. Include these files as Appendices A, B, C, and D with this completed form.

4.3.2 Complete The Test Procedure Form

Ensure that all blank fields in this report are completed correctly and submit the completed report to QA.
SUMMARY SHEET

OPERATION TITLE: _____________________________ WOA# ____________

TEST ARTICLES IDENTIFICATION (including serial and/or part numbers):

____________________________________________________________________

TASKS/STEPS COMPLETED: ___________________________________________

____________________________________________________________________

LOCATION: _________________________________________________________

TEST STARTED: TEST TERMINATED

TIME: _______ Hr/Min TIME: _______ Hr/Min

DATE: _______ DATE: _______

LOGS USED: ________________________________________________________

ANOMALY REPORTS GENERATED: ___________________________________

____________________________________________________________________

COMMENTS: ________________________________________________________

____________________________________________________________________

____________________________________________________________________

____________________________________________________________________

____________________________________________________________________

TEST CONDUCTOR: _________________________________________________

Signature/Date

QA REPRESENTATIVE: ________________________________________________

Signature/Date
Appendix A.  Shell Script u

#!/bin/sh
pkill cosnoopy
perl ../UniScript.pl stp5_2_3_1 "C000,C100,0330,2460,0,0,0,0"
cosnoopy&
Appendix B. Test Script stp5_2_3_1.tst

;*********************************************************************
;*********************************************************************
;** ***
;** BBBB OOO OOO TTTTT OOO N N L Y Y **
;** B B O O O O T O O N N L Y Y **
;** BBBB O O O O O T O O N N N L Y Y **
;** B B O O O O T O O N N N L Y Y **
;** BBBB O O O O O T O O N N L Y Y **
;** BBBB OOO OOO T OOO N N LLLLL Y **
;**
;*********************************************************************
;*********************************************************************
;*********************************************************************
;STP 5.2.3.1
;------------------------------------------------------------------------------*
* Verify that the HK packets produced in the test of 5.2.2.1 above have less *
* than 1 second intervals between each packet. *
*------------------------------------------------------------------------------*
* The script makes rather opaque use of two Perl variables, $xt and $yt. *
* $xt is initialized by means of a CHECK directive that always succeeds, namely *
* CHECK 1,((($xt=$LFCTIME+$LFDMONS[7]/50.0)==$xt)
* * After each HK packet is received, the following operations are performed *
* *
* $yt = $LFCTIME+$LFDMONS[7]/50.0; # seconds + (fiftieths of a second)/50 *
* if ($yt < $xt + 0.9) # this should not be later than the last *
* {$xt = $yt} # t-value OK, update $xt *
* else *
* {$xt = 0} # if not, set $xt=0; CHECK will discover *
* if ($xt == 0) *
* {terminate the script}
* *
* These operations are, somewhat obliquely, compressed into a single CHECK *
* directive as follows:
* CHECK 1,((($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<= $xt+.9) ? $yt : 0) > 0)
* * Parameters:
* #0 SOURCE ("scratch area")
* #1 DEST ("scratch area + 256")
* #2 NOOP (address of LFDNOOP command routine)
* #3 mTICKS (address of mTICKS in Patchable Constants)
;*********************************************************************
;ECHO 2
;SYM SOURCE =0x#0
SYM DEST =0x#1
SYM NOOP =0x#2
;SYM NBYTES =16
SYM NBYTES =1024
SYM CODE =0
SYM ID =0
SYM SETTING =0
SYM BANK =0
SYM RATE =0
SYM SEGMENT =0
SYM DIR =0
SYM POWER =0
SYM STATE =0
SYM HIVOLT =0
SYM LIMIT =0
SYM VOLTAGE =0
SYM ACTUATOR=0
SYM ABORT =0
SYM OVERRIDE=0
SYM DOOR =0
SYM MOVE =0
SYM BANK =0
SYM DATA =0
SYM DELTA1 =5
SYM DELTA2 =25
SYM NSC =5
SYM EXTERN =0
SYM mTICKS =0x#3

; *****************************************
; * Wait until setup (if any) is complete *
; *****************************************

; WTOR "Press Y when ready to conduct test 5.2.3.1"

; **********************************************
; * Force Boot State, set up monitor for mTICKS *
; **********************************************

; DTG 3,"(0) Sending POR, collecting initial HK"
WTOR "Sending POR, collecting initial HK"
POR
;DELAY DELTA1
WAIT 2
LFDMADDR 7,mTICKS,EXTERN
WAIT NSEC,HK
LOG 1,LFDCMDX,LFDCMDR,LFCPKT,LFDCBUF,LFCTIME,LFDMONS

; *********************************************
; * Sync up with HK, get initial value of $xt *
; *********************************************

; LFDNOOP

; DTG 3,"(1) Sending LFDNOOP to get command stream started"
WTOR "Sending LFDNOOP to get command stream started"
CHECK 1,((xt=$(LFCTIME+$LFDMONS[7]/50.0)==$xt)
LFDNOOP

; DELAY DELTA1
WAIT 0,HK
LOG 1,LFDCMDX,LFDCMDR,LFCPKT,LFDCBUF,LFCTIME,LFDMONS
CHECK 1,((xt=((yt=$(LFCTIME+$LFDMONS[7]/50.0)==$(xt+.1))*($yt:0)>0)

; DTG 3,"(2) Sending LFDCOPY"
WTOR "Sending LFDCOPY"
LFDCOPY SOURCE,DEST,NBYTES,BANK
LFDCOPY SOURCE,SOURCE,NBYTES,BANK
DELAY DELTA1
WAIT 0,HK
LOG 1,LFDCMDX,LFDCMDR,LFCPKT,LFDCBUF,LFCTIME,LFDMONS
CHECK 1,((xt=((yt=$(LFCTIME+$LFDMONS[7]/50.0)==$(xt+.1))*($yt:0)>0)

; DTG 3,"(3) Sending LFMCRC"
WTOR "Sending LFMCRC"
LFDCRC SOURCE,NBYTES,CODE
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(4) Sending LFDDIAGC"
WTO "Sending LFDDIAGC"
LFDDIAGC
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(5) Sending LFDDNLOD"
WTO "Sending LFDDNLOD"
LFDDNLOD SOURCE, NBYTES
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(6) Sending LFDGOTO"
WTO "Sending LFDGOTO"
LFDGOTO NOOP
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(7) Sending LFDHKREQ"
WTO "Sending LFDHKREQ"
LFDHKREQ
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(8) Sending LFDMADDR"
WTO "Sending LFDMADDR"
LFDMADDR 0, SOURCE, DATA
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(9) Sending LFDUPLOD"
WTO "Sending LFDUPLOD"
LFDUPLOD DEST, NBYTES, 0
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(10) Sending LFDWDOG"
WTO "Sending LFDWDOG"
LFDWDOG 1
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(11) Sending LFDFNOOP"
WTO "Sending LFDFNOOP"
LFDFNOOP
DELAY DELTA1
WAIT 0, HK
LOG 1, LFDCMDX, LFDCMDR, LFCPKT, LFDCBUF, LFCTIME, LFDMONS
CHECK 1, (($xt=((($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
;
DTG 3, "(12) Sending LFDFMPCS"
;WTO       "Sending LFDJMPCS"
;LFDJMPCS  0
;LFGWBK    SETTING,SEGMENT,0
;LFGREWK   SETTING,SEGMENT,0
;LFGLEQT   SETTING,SEGMENT
;LFGSHFT   SETTING,SEGMENT,0
;LFGSTIM   RATE,SEGMENT
;LFGSTR    SETTING,SEGMENT,0
;LFGTT     SETTING,SEGMENT,DIR
;LFGUQT    SETTING,SEGMENT
;LFGQPWR   0
;LFHRAMPT  RATE
;LFHSTATE  STATE
;LFHVENA   HINVOLT
;LFHVILIM  LIMIT
;LFHVLOW   VOLTAGE,SEGMENT
;LFHVMAX   VOLTAGE,SEGMENT
;LFHVNOM   VOLTAGE,SEGMENT
;LFHVPR    POWER
;LFHVSET   VOLTAGE,0
;LFPCRP    INTERVAL,SEGMENT,COUNT
;LFRACT1   POWER
;LFRACT2   POWER
;LFRACTEN  ACTUATOR
;LFRACTRS  0
;LFRAFPR   POWER
;LFRLIM    LIMIT
;LFROVDIR  OVERRIDE
;LFROMDIR  DIR
;LFROSNA   DOOR
;LFROFPR   MOVE
;
;WAIT      NSEC,HK
;LOG       1,LFDCBUF,LFCTIME,LFDCMDX,LFDCMDR,LFCPKT
;CHECK     1,($LFDMDR==LFCPKT && LFDCMDX==9)
;
;DTG       1, "(12) Test stp5.2.3.1 completed successfully"
WTO       "Test stp5.2.3.1 completed successfully"
Appendix C. Test Report stp5_2_3_1.rp1

Ver 01.09 Fri Nov 17 05:03:54 2000 "(0) Sending POR, collecting initial HK"

LFDMADDR 7,mTICKS,EXTERN

Addr Addr HK-Name Value
---- ---- -------------- ----- 
170C-170D LFDCMDX 0003
1718-1719 LFDCMDR 0003
1700-1703 LFCPKT 0000007F
1664-167F LFDCBUF 8080 7F7F 0042 FFBD 0000 FFFF 0000 FFFF 0000 FFFF 0000
FFFF 0000 FFFF
1680-1683 LFCTIME 0000009D
1738-173F LFDMONS 00 00 00 00 00 00 00 00

Ver 01.09 Fri Nov 17 05:03:56 2000 "(1) Sending LFDNOOP to get command stream started"

CHECK: (((xt=$LFCTIME+$LFDMONS[7]/50.0)==$xt)
eval: ((0000=009D+0000[7]/50.0)==0000)
SUCCESS
LFDNOOP

Addr Addr HK-Name Value
---- ---- -------------- ----- 
170C-170D LFDCMDX 0001
1718-1719 LFDCMDR 0001
1700-1703 LFCPKT 00000001
1664-167F LFDCBUF 8181 7E7E 0001 FFFE 0007 FFF8 2460 DB9F 0000 FFFF 0000
FFFF 0000 FFFF
1680-1683 LFCTIME 00000001
1738-173F LFDMONS 00 00 00 00 00 00 00 2F

CHECK: (((xt=($LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
eval: ((009D=((0000=0001+0000[7]/50.0)<=009D+.9)70000.0)>0)
SUCCESS

Ver 01.09 Fri Nov 17 05:03:56 2000 "(2) Sending LFDCOPY"

LFDCOPY SOURCE, SOURCE, NBYTES, BANK

Addr Addr HK-Name Value
---- ---- -------------- ----- 
170C-170D LFDCMDX 0001
1718-1719 LFDCMDR 0001
1700-1703 LPFCKT 00000002
1664-167F LFDCBUF 8080 7F7F 0002 FFFD 0000 FFFF 0000 FFFF 0000 FFFF 0000 FFFF 0000 FFFF
1680-1683 LFCTIME 00000002
1738-173F LFDMONS 00 00 00 00 00 00 00 0B
CHECK: (($xt=(($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
eval: ((0001=((0001=0002+0000[7]/50.0)<=0001+.9)?0001:0)>0)
SUCCESS
Ver 01.09 Fri Nov 17 05:03:57 2000 "(3) Sending LFMCRC"

LFDCRC SOURCE,NBYTES,CODE
Addr Addr HK-Name Value
---- ---- -------------- ----- 170C-170D LFDCMDX 0002
1718-1719 LFDCMDR 0002
1700-1703 LPFCKT 00000003
1664-167F LFDCBUF 8383 7C7C 0003 FFFC C000 3FFF C000 3FFF 0400 FBFF 0000 FFFF 0000 FFFF
1680-1683 LFCTIME 00000002
1738-173F LFDMONS 00 00 00 00 00 00 00 1C
CHECK: (($xt=(($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
eval: ((0002=((0002=0002+0000[7]/50.0)<=0002+.9)?0002:0)>0)
SUCCESS
Ver 01.09 Fri Nov 17 05:03:57 2000 "(4) Sending LFDDIAGC"

LFDDIAGC
Addr Addr HK-Name Value
---- ---- -------------- ----- 170C-170D LFDCMDX 0003
1718-1719 LFDCMDR 0003
1700-1703 LPFCKT 00000004
1664-167F LFDCBUF 8282 7D7D 0004 FFBF C000 3FFF 0400 FBFF 0000 FFFF 0000 FFFF
1680-1683 LFCTIME 00000002
1738-173F LFDMONS 00 00 00 00 00 00 00 2A
CHECK: (($xt=(($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0)
eval: ((0002=((0002=0002+0000[7]/50.0)<=0002+.9)?0002:0)>0)
SUCCESS
Ver 01.09 Fri Nov 17 05:03:57 2000 "(5) Sending LFDDNLOD"

LFDDNLOD SOURCE,NBYTES
Addr Addr HK-Name Value
---- ---- -------------- ----- 170C-170D LFDCMDX 0003
1718-1719 LFDCMDR 0003
1700-1703 LPFCKT 00000004
<table>
<thead>
<tr>
<th>Address</th>
<th>HK-Name</th>
<th>Value</th>
<th>Address</th>
<th>HK-Name</th>
<th>Value</th>
<th>Address</th>
<th>HK-Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1664-167F LFDCBUF</td>
<td>8282 7D7D 0004 FFFB C000 3FFF 0400 FBFF 0000 FFFF 0000 FFFF 0000 FFFF 0000</td>
<td>1738-173F LFDMONS</td>
<td>00 00 00 00 00 00 00 2A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1680-1683 LFCTIME</td>
<td>00000002</td>
<td>1738-173F LFDMONS</td>
<td>00 00 00 00 00 00 00 2A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHECK: (($xt=(($yt=$LFCTIME+$LFDMONS[7]/50.0)&lt;=$xt+.9)?$yt:0)&gt;0)</td>
<td>eval: (($0002=(($0002=0000+0000[7]/50.0)&lt;=$0002+.9)?$0002:0)&gt;0)</td>
<td>SUCCESS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ver 01.09 Fri Nov 17 05:03:57 2000 "(6) Sending LFDGOTO" LFDGOTO NOOP

Addr Addr HK-Name Value
----- ----- --------------- -------
170C-170D LFDCMDX 0005
1718-1719 LFDCMDR 0005
1700-1703 LFCKPT 00000006
1664-167F LFDCBUF AEAE 5151 0006 FFF9 C000 3FFF 0400 FBFF 0000 FFFF 0000 FFFF 0000 FFFF 0000 | 1680-1683 LFCTIME | 00000003 |
| 1738-173F LFDMONS | 00 00 00 00 00 00 00 1D |
| CHECK: (($xt=(($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0) | eval: (($0003=((0003=0003+0000[7]/50.0)<=0003+.9)?$0003:0)>0) | SUCCESS |

Ver 01.09 Fri Nov 17 05:03:58 2000 "(7) Sending LFDHKREQ" LFDHKREQ

Addr Addr HK-Name Value
----- ----- --------------- -------
170C-170D LFDCMDX 0005
1718-1719 LFDCMDR 0005
1700-1703 LFCKPT 00000007
1664-167F LFDCBUF EAEA 5151 0007 FFF8 0330 FCCF 0000 FFFF 0000 FFFF 0000 FFFF 0000 FFFF 0000 | 1680-1683 LFCTIME | 00000003 |
| 1738-173F LFDMONS | 00 00 00 00 00 00 00 28 |
| CHECK: (($xt=(($yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?$yt:0)>0) | eval: (($0003=((0003+0003+0000[7]/50.0)<=0003+.9)?$0003:0)>0) | SUCCESS |

Ver 01.09 Fri Nov 17 05:03:58 2000 "(8) Sending LFDMADDR" LFDMADDR 0,SOURCE,DATA

Addr Addr HK-Name Value
----- ----- --------------- -------
170C-170D LFDCMDX 0006
1718-1719 LFDCMDR 0006
1700-1703 LFCKPT 00000008
1664-167F LFDCBUF FFFF 0000 0008 FFF7 0000 FFFF 0000 FFFF 0000 FFFF 0000 FFFF 0000 FFFF 0000 | 1680-1683 LFCTIME | 00000004 |
### Requirement 5.2.3.1 Housekeeping Response Within One Second

#### Component Test Results

**CHECK:** \( ((\$xt=((\$yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?)?0+y) 0) > 0) \)

**eval:** \( (((0003+(0003+0000+0000[7]/50.0)<=0003+.9)?0003:0)>0) \)

---

**SUCCESS**

**Ver 01.09 Fri Nov 17 05:03:58 2000**  
"(9) Sending LFDUPLOD"

<table>
<thead>
<tr>
<th>Addr</th>
<th>Addr</th>
<th>HK-Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170C-170D</td>
<td>LFDCMDX</td>
<td>0007</td>
<td></td>
</tr>
<tr>
<td>1718-1719</td>
<td>LFDCMDR</td>
<td>0007</td>
<td></td>
</tr>
<tr>
<td>1700-1703</td>
<td>LFCPKT</td>
<td>00000009</td>
<td></td>
</tr>
</tbody>
</table>

**Ver 01.09 Fri Nov 17 05:03:59 2000**  
"(10) Sending LFDWDOG"

<table>
<thead>
<tr>
<th>Addr</th>
<th>Addr</th>
<th>HK-Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170C-170D</td>
<td>LFDCMDX</td>
<td>0008</td>
<td></td>
</tr>
<tr>
<td>1718-1719</td>
<td>LFDCMDR</td>
<td>0008</td>
<td></td>
</tr>
<tr>
<td>1700-1703</td>
<td>LFCPKT</td>
<td>0000000A</td>
<td></td>
</tr>
</tbody>
</table>

**Ver 01.09 Fri Nov 17 05:03:59 2000**  
"(11) Sending LFDNOOP"

<table>
<thead>
<tr>
<th>Addr</th>
<th>Addr</th>
<th>HK-Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170C-170D</td>
<td>LFDCMDX</td>
<td>0009</td>
<td></td>
</tr>
<tr>
<td>1718-1719</td>
<td>LFDCMDR</td>
<td>0009</td>
<td></td>
</tr>
<tr>
<td>1700-1703</td>
<td>LFCPKT</td>
<td>0000000B</td>
<td></td>
</tr>
</tbody>
</table>

---

**CHECK:** \( ((\$xt=((\$yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?)?0+y) 0) > 0) \)

**eval:** \( (((0004+(0004+0004+0000[7]/50.0)<=0004+.9)?0004:0)>0) \)

---

**SUCCESS**

**Ver 01.09 Fri Nov 17 05:03:59 2000**  
"(10) Sending LFDWDOG"

<table>
<thead>
<tr>
<th>Addr</th>
<th>Addr</th>
<th>HK-Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170C-170D</td>
<td>LFDCMDX</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>1718-1719</td>
<td>LFDCMDR</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>1700-1703</td>
<td>LFCPKT</td>
<td>00000001</td>
<td></td>
</tr>
</tbody>
</table>

**Ver 01.09 Fri Nov 17 05:03:59 2000**  
"(11) Sending LFDNOOP"

<table>
<thead>
<tr>
<th>Addr</th>
<th>Addr</th>
<th>HK-Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170C-170D</td>
<td>LFDCMDX</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>1718-1719</td>
<td>LFDCMDR</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>1700-1703</td>
<td>LFCPKT</td>
<td>00000002</td>
<td></td>
</tr>
</tbody>
</table>

**CHECK:** \( ((\$xt=((\$yt=$LFCTIME+$LFDMONS[7]/50.0)<=$xt+.9)?)?0+y) 0) > 0) \)
eval:  ((0004=((0004=0005+0000[7])/50.0)<=0004+.9)?0004:0)>0)
SUCCESS
Ver 01.09 Fri Nov 17 05:03:59 2000  "(12) Test stp5.2.3.1 completed successfully"
Appendix D. Test Report stp5_2_3_1.rp2

Ver 01.09 Fri Nov 17 05:03:54 2000 "(0) Sending POR, collecting initial HK"
---------------------------------------------------
POR PACKET
---------------------------------------------------
80000000
---------------------------------------------------
COMMAND PACKET
---------------------------------------------------
<table>
<thead>
<tr>
<th>PARM4</th>
<th>PARM3</th>
<th>PARM2</th>
<th>PARM1</th>
<th>PARM0</th>
</tr>
</thead>
<tbody>
<tr>
<td>045AFFFE</td>
<td>04580000</td>
<td>0456FFFF</td>
<td>04540000</td>
<td>0452FFFF</td>
</tr>
<tr>
<td>SN</td>
<td>OPCODE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0446FFFE</td>
<td>04440000</td>
<td>04427E7E</td>
<td>04408181</td>
<td></td>
</tr>
</tbody>
</table>
---------------------------------------------------
Ver 01.09 Fri Nov 17 05:03:56 2000 "(1) Sending LFDNOOP to get command stream started"
---------------------------------------------------
COMMAND PACKET
---------------------------------------------------
<table>
<thead>
<tr>
<th>PARM4</th>
<th>PARM3</th>
<th>PARM2</th>
<th>PARM1</th>
<th>PARM0</th>
</tr>
</thead>
<tbody>
<tr>
<td>045AFFFE</td>
<td>04580000</td>
<td>0456FFFF</td>
<td>04540000</td>
<td>0452FFFF</td>
</tr>
<tr>
<td>SN</td>
<td>OPCODE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0446FFFD</td>
<td>04440000</td>
<td>04427F7F</td>
<td>04408080</td>
<td></td>
</tr>
</tbody>
</table>
---------------------------------------------------
Ver 01.09 Fri Nov 17 05:03:56 2000 "(2) Sending LFDCOPY"
---------------------------------------------------
COMMAND PACKET
---------------------------------------------------
<table>
<thead>
<tr>
<th>PARM4</th>
<th>PARM3</th>
<th>PARM2</th>
<th>PARM1</th>
<th>PARM0</th>
</tr>
</thead>
<tbody>
<tr>
<td>045AFFFE</td>
<td>04580000</td>
<td>0456FFFF</td>
<td>04540000</td>
<td>0452FFBF</td>
</tr>
<tr>
<td>SN</td>
<td>OPCODE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0446FFFC</td>
<td>04440000</td>
<td>04427C7C</td>
<td>04408383</td>
<td></td>
</tr>
</tbody>
</table>
---------------------------------------------------
Ver 01.09 Fri Nov 17 05:03:57 2000 "(3) Sending LFMCRC"
Ver 01.09 Fri Nov 17 05:03:57 2000  "(4) Sending LFDDIAGC"

Ver 01.09 Fri Nov 17 05:03:57 2000  "(5) Sending LFDDNLOD"

Ver 01.09 Fri Nov 17 05:03:57 2000  "(6) Sending LFDGOTO"

Ver 01.09 Fri Nov 17 05:03:58 2000  "(7) Sending LFDHKREQ"

Ver 01.09 Fri Nov 17 05:03:58 2000  "(8) Sending LFDMADDR"
<table>
<thead>
<tr>
<th>SN OPCODE</th>
<th>0446FFF6 04440009 04427E7E 04408181</th>
</tr>
</thead>
</table>

Ver 01.09 Fri Nov 17 05:03:58 2000 "(9) Sending LFWDUPLOD"

<table>
<thead>
<tr>
<th>COMMAND PACKET</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARM4</td>
</tr>
<tr>
<td>045AFFFF</td>
</tr>
</tbody>
</table>

SN OPCODE 0446FFF5 0444000A 04425252 0440ADAD

Ver 01.09 Fri Nov 17 05:03:59 2000 "(10) Sending LFWDWDOG"

<table>
<thead>
<tr>
<th>COMMAND PACKET</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARM4</td>
</tr>
<tr>
<td>045AFFFF</td>
</tr>
</tbody>
</table>

SN OPCODE 0446FFF4 0444000B 04420E0E 0440F1F1

Ver 01.09 Fri Nov 17 05:03:59 2000 "(11) Sending LFWDNOOP"

<table>
<thead>
<tr>
<th>COMMAND PACKET</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARM4</td>
</tr>
<tr>
<td>045AFFFF</td>
</tr>
</tbody>
</table>

SN OPCODE 0446FFF3 0444000C 04427F7F 04408080