**W.W. Hansen Experimental Physics Laboratory**
**STANFORD UNIVERSITY**
**STANFORD, CA 94305 – 4085**

## Gravity Probe B Relativity Mission

# Operational Procedure for Baseline and Regression Testing of TDP

## P0949 Rev D

**2/26/03**

## Approvals

| NAME | SIGNATURE | DATE |
|---|---|---|
| Jennifer Spencer<br>*Data Processing Lead* | | |
| Paul McGown<br>*Chief Bitsmith* | | |
| Ron Sharbaugh<br>*S/W Manager* | | |
| Marcie Smith<br>*MOC Project Manager* | | |
| Kelly Burlingham<br>*Software Quality Assurance* | | |

## Required Signatures prior to Execution

| NAME | SIGNATURE | DATE |
|---|---|---|
| NAME:<br>*Test Engineer* | | |
| Kelly Burlingham<br>*Software Quality Assurance* | | |

Tom Langenstein ITAR Assessment Performed, ITAR Control Req'd?

___ Yes ___ No

______________________________________________

## Table of Contents:

# 1 REVISION HISTORY

| REV | DATE | AUTHOR | COMMENTS |
|---|---|---|---|
| - | 16 Oct 2002 | JLM | initial version |
| A | 30 May 2003 | JLS | Added baseline testing to this document, changed it from a regression-test-only document to a full test plan. |
| B | 9 Sept 2003 | JLS | Section 10.1 updated (TDP5) to reflect new data tables. |
| C | 11 Nov 2003 | JLS | Re-arranged order of test sections to reflect actual order of tests. No content changes. |
| D | 26 Feb 2004 | JLS | Updated section 10.5 in response to code changes. |

## 2 SCOPE

This Test Plan Document details the TDP software package and how to both baseline and regression test it.

## 3 OPERATIONAL PERSONNEL

Jennifer Spencer
Paul McGown
Qualified QA Rep: Kelly Burlingham

## 4 REQUIREMENTS & CONSTRAINTS

### 4.1 Hardware and Software Requirements

Operations are performed on the Sun server machine known as "science". If spacecraft or simulated data is required, it will be from either a data tape delivered from the Lockheed Martin Integrated Test Facility (ITF) or an FTP'd data file from the ITF or a POD.

### 4.2 Configuration Requirements

The operator must be logged into the server "science" as the user "tdp" and must know the MSS Version of any data they are using for test purposes. The user should reserve a directory for test data and save this data for QA.
Directory: ______________________________________________

### 4.3 Constraints

| Constraint | Risk |
|---|---|
| Time required for regression testing. | At least three hours is required for regression testing and considerably more for baseline testing. At this time, data processing is not directly impacted but support for the department may be unavailable due to personnel working tests. |
| Quality of VC1and2 or VC0 binary data | Exceptionally good quality data could perhaps not fully test TDP's ability to filter out bad packets. Recommend using data known to be poor quality over exceptionally good quality based on risk listed above. Bad data includes: gaps, bad packet lengths, negative time ranges, data with values of floating point infinity and negative zero, for example. |

## 5 REFERENCE DOCUMENTS

| Document | Document No. | ALIAS. |
|---|---|---|
| Data Management Plan | S0331 | |
| Stanford Post-Processing Operations for Science Mission Data | S0401 | |
| MOC Configuration Control, IONET LAN | S0476 | |
| LM VDD: Real-time Ground Software | LMMS/P479910O | |
| IONET_CONF | | |
| Telemetry Data Processing "Level 0" Verification | S0572 | |
| Telemetry Data Processing "Level 1" Verification | S0603 | |
| TDP/TCAD Software Release (Design Document) | S0613 | |

| | | |
|---|---|---|
| MOC Requirements | MO-02 | |
| Processing Data from High-resolution to an Averaged Data Set | P0905 | |
| Event Display in Non-Real-Time System | P0832 | |
| DBRO Data Retrieval and Display | S0638 | |
| MOC VDD for EVENTget-pl | S0515 | |

## 6 QUALITY ASSURANCE PROVISIONS

Quality Assurance must be given 24 hour notification before this test is run; presence is at their discretion.

QA Notified Date & Time: ___________________ By: _________________________ QA Initials: _______

## 7 TEST ENVIRONMENT

This software is tested on the science server at SU under the following configuration:

| Software Configurations | Version Number (fill in) |
|---|---|
| IDL | |
| Sybase | |
| Solaris | |
| MSS | |
| PERL | |

## 8 TEST CASES FOR TDP GENERAL ROUTINES

Test case components are listed in the table below, along with whether or not they need to be run and if so, initials of who has run the tests. Enter either B for baseline testing or R for regression testing under type of verification required. The objective for regression testing each component is always to get a zero difference when comparing the old revision's output files (from the old, production software directory) to those from the new revision (in the development directory awaiting release). Regression tests are considered passed if no differences are found in running binary or ASCII differences between old files and new files. If there are differences, or if code module is completely new to this release, baseline testing must be done. Any data output differences (as opposed to simple report formatting differences) must be explained by updated design documentation. Pass/Fail criteria for each test are set forth below, and supercede any general instructions for specific test cases. Test set up for final verification of software requires that no further modifications will be made to code and that no code is checked out of RCS for editing. Execution procedures and result descriptions follow each case below.

If the code in any of the following files has changed since the previous software release, the tests in the corresponding section number must be run and checked off for verification.

LASP version being tested: ___________________________________

| **VERIFICATION REQUIRED**<br>• **Baseline (B)**<br>• **Regression (R)** | **FILE (under lasp-x.x/src)** | **UNIX date & filesize on program directory** | **TEST NAME** | **TEST SECTION** | **VERIFIED BY** *(initials)* | **DATE** |
|---|---|---|---|---|---|---|
| | Throughput/Speed (no filename - use src/tdp directory) /apps/supported/lasp/tdp | | TDP1 | Section 8.1 | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | MSS Import src/db/load_x_x_x directory | | TDP10 | Section 8.2 | | |

## 8.1 TDP1: Throughput/Speed

❑ (B) ❑ (R) ______ initials

Test Case Verification Number: TDP1

INTRODUCTION
This test case is used for doing timing measurements of TDP processing.

APPROACH
Simply time how long process per P0826 or P0905 (depending on whether this is testing raw file → final data processing or from L0 → final data processing) takes.

FEATURES TO BE TESTED (CHECK ONE)

❑ How long “standard” importation of VC files (spacecraft binary files) from a GN pass takes (requirement MOC-18) to process through Level 1.

❑ How long retrieval from L0 to L1 takes (requirement MOC-19).

FEATURES NOT TO BE TESTED
- Accuracy of processing
- File splicing in the event of clock resets
- Data retrieval speeds
- Extra features such as limit checking, GPS processing or Snapshot building, which are only done when those data types are found in the data file.
- Does not include any time required for SAFS data transfer through IOnet.

TESTS

❑ Log the wall-clock time or UNIX “date” time at start and end of running P0826 or read the on-screen data summary which contains the time for processing.

❑ Log the wall-clock time or UNIX “date” time at start and end of running P0905 or read the on-screen data summary which contains the time for processing.

PASS/FAIL
For MOC-18/19, the duration times must be less than spec.

**RESULT:** PASS FAIL (circle one) ______ initials

## 8.2 TDP10: MSS Database Import

❑ (B) ❑ (R) ______ initials

Test Case Verification Number: TDP10

INTRODUCTION

To ensure that the latest MSS database from Lockheed is imported properly onto the "Science" server at Stanford's Gravity Probe B Mission Operations Center. This database is used by TDP for decommutating the telemetry from the raw spacecraft files.

APPROACH
Visual inspection of output files to be imported to Sybase.

FEATURES TO BE TESTED
- Check that files to be imported to Level 1 metadata are as expected from the populate scripts

FEATURES NOT TO BE TESTED
- Integrity of MSS database

TESTS

❑ BASELINE
Run the MSS import process as described in P0908 (MSS to TDP/TCAD Database Population Process) through step 11.19. During this step, the shell script will ask the user to hit any key to continue. At this point, halt the process by typing Ctrl+C and inspect the output from the populate shell script. When inspection is complete, inspect the data sets that would be produced by the populate_tm*.sql scripts by running the procedure on Sybase command line to just before the "execute" state. Inspect results.

PASS/FAIL
Data sets to be imported pass visual inspection
The populate shell script created proper scripts and directories.

**RESULT:** PASS FAIL (circle one) _____ initials

REGRESSION → SEE RESULTS FOR TDP5
If data passes regression or baseline test for TDP5 (Level 1 processing), then the MSS database import was necessarily correct. If the data passes L1 verification then the MSS import was done properly and the MSS testing is working. Regression testing of this routine is not applicable.

While not required to show proof of success, if desired, a "regression" test of each MSS import could be performed by following the steps in P0908 and by then comparing the log files generated by the import process against the LM VDD deliverable document.

# 9 TEST CASES FOR LEVEL 0

Test case components are listed in the table below, along with whether or not they need to be run and if so, initials of who has run the tests. Enter either B for baseline testing or R for regression testing under type of verification required. The objective for regression testing each component is always to get a zero difference when comparing the old revision's output files (from the old, production software directory) to those from the new revision (in the development directory awaiting release). Regression tests are considered passed if no differences are found in running binary or ASCII differences between old files and new files. If there are differences, or if code module is completely new to this release, baseline testing must be done. Any data output differences (as opposed to simple report formatting differences) must be explained by updated design documentation. Pass/Fail criteria for each test are set forth below, and supercede any general instructions for specific test cases. Test set up for final verification of software requires that no further modifications will be made to code and that no code is checked out of RCS for editing. Execution procedures and result descriptions follow each case below.

If the code in any of the following files has changed since the previous software release, the tests in the corresponding section number must be run and checked off for verification.

LASP version being tested: ________________________________

| **VERIFICATION REQUIRED** • **Baseline (B)** • **Regression (R)** | **FILE (under src/tdp)** | **UNIX date & filesize on program file** | **TEST NAME** | **TEST SECTION** | **VERIFIED BY** *(initials)* | **DATE** |
|---|---|---|---|---|---|---|
| | tdp start script /lasp/scripts/tdp | | TDP13 | Section 9.1 | | |
| | IPDUsplice.pro | | TDP11 | Section 9.2 | | |
| | gpb_tdp_L0.pro | | TDP2 | Section 9.3 | | |
| | process_tcor.pro | | TDP3 | Section 9.4 | | |
| | ssbuild.pro | | TDP4 | Section 9.5 | | |
| | l02l1 | | TDP6 | Section 9.6 | | |
| | L0_to_L1_display.pro | | TDP6 | Section 9.6 | | |
| | L0_to_L1.pro | | TDP6 | Section 9.6 | | |

## 9.1 TDP13: TDP start script

❑ (B) (regression test not applicable) _____ initials

Test Case Verification Number: TDP13

❑ BASELINE

INTRODUCTION
This test verifies whether the TDP startup script works.

FEATURES TO BE TESTED
– Properly start TDP and use the proper release version

FEATURES NOT TO BE TESTED
– None

TEST
The following requires a set of VC files that contain all packet types (this can be just one file). Each file is a different test:

i. Run “tdp” at the command line of any science terminal.
ii. This should start /apps/supported/scripts/tdp, which in turn is symbolically linked to /apps/supported/lasp/scripts/tdp.
iii. Inspect the output from the command and read the version number

PASS/FAIL
TDP started using the proper code release version.

**RESULT:** PASS FAIL (circle one) _____ initials

## 9.2 TDP11: IPDU frame parsing on raw files

❑ (B) ❑ (R) ______ initials

Test Case Verification Number: TDP11

INTRODUCTION

TDP processes one binary spacecraft file at a time (the “raw” data). One file represents one pass or “dump” from the spacecraft taken during a pass and shipped to the Mission Operations Center from SAFS. Each binary file is composed of blocks of data known as “IPDUs” (Internet Protocol Data Units). Each IPDU is made up of a satellite transfer frame (of a known, constant size) and a header (the IPDU header - also a constant size and format).

IPDU headers are assigned to each data packet by the Front-End Processor (FEP) at the receiving ground station. These IPDU headers essentially describe what time the data arrived on the ground and some Reed-Solomon processing status information.

❑ BASELINE

APPROACH
IPDUsplice.pro is a difficult program to test. It involves a great deal of inspection against original specifications of data assembly found in SCSE-16, section 9.

FEATURES TO BE TESTED
- Correct packets are parceled out and labeled as correct IPDUs
- Bad data is removed
- Good data is not thrown out with the bad
- SSR data wraps taken into account
- SV clock resets to zero taken into account
- SV clock “advances” or “backups” of more than one day are taken into account

FEATURES NOT TO BE TESTED
- None

TESTS

i. Using the specification for an IPDU header and the definition of a valid spacecraft data packet from SCSE-16, section 9, one must inspect all data rejected as “bad” by IPDUsplice and confirm that it is indeed bad data.
ii. Then, run a good, completely clean file through IPDUsplice and the result should be only one splice – the original file contents. Do a binary difference between the original file and the splice; there should be no differences.
iii. For confidence, cycle through the data labeled as good by IPDUsplice. Inspect it visually using an octal dump (od –tx1).

PASS/FAIL
IPDUsplice files are successfully inspected.

**RESULT:** PASS FAIL (circle one) ______ initials

❑ REGRESSION

TEST

IPDUsplice.pro outputs should be the same between lasp-x.x versions. If the program has been altered, run the released version and the new new version of IPDUsplice on a VC*.bin file in two test directories. Do a binary comparison of the binary output files of this tool (the text files may differ, but they are summaries of information only – essential content should be the same and should be inspected) between the old version of the code and the new version (binary diff as detailed in TDP2 above). Any differences should be explained by documentation and a baseline test should be considered.

PASS/FAIL
Each comparison must yield no difference between released and new code output files.

**RESULT:** PASS FAIL (circle one) ______ initials

## 9.3 TDP2: Packet Parsing/Decomposition

❑ (B) ❑ (R) ______ initials

Test Case Verification Number: TDP2

INTRODUCTION
This test verifies the accuracy of packet decomposition.

❑ BASELINE
Specifically, it checks that the content of the output file of SU's TDP (to be bcp'd into Sybase as the Level 0 product) matches that analyzed by LM's EDR (which is a verified deliverable CSCI).

APPROACH
A verification program, gpb_tdp_L0.pro, is developed to exactly duplicate both TDP and EDR processing – i.e. one inputs a VC file, and this third-party program outputs either a file in the format of an SU TDP binary file or in the format of an LM EDR binary file. Note that EDR processing rejects NO packets, while gpb_tpb_L0.pro discards any packet that it can detect as bad (such as date stamps over 2006, improper packet ID or length, et cetera).
The same input VC file(s) is processed four times to verify each APID type:

i. Once by SU TDP.
ii. Once by LM EDR.
iii. Once by the verification program to output SU TDP files.
iv. Once by the verification program to output LM EDR files.

For each VC input file, the output TDP format files are compared, and then the output EDR format files are compared.

FEATURES TO BE TESTED
– Properly parse VC (IPDU) file into spacecraft packets.
– Properly assessing and categorizing packet types.
– Using proper packet lengths (per packet type).
– Assigning proper packet timestamps for each packet (only packets 100 and 301 receive timestamps from the vehicle; all others must use the p100 times within the same transfer frame).

FEATURES NOT TO BE TESTED
– Rejection of bad data is not directly tested. It is tested through regression testing (see P0949) instead. New, better filters and the final data processing summary show direct, verifiably bad pieces as "leftovers" during a before & after file comparison between code versions.

TEST
The following requires a set of VC files that contain all packet types (this can be just one file). Each file is a different test:

i. Run the programs to create the 4 different output files.

ii. Compare the TDP output file (i above) with the test program “tdp format” output file (iii above).
iii. Compare the EDR output file (ii above) with the test program “EDR format” output file (iv above).

PASS/FAIL
Each comparison must be identical.

**RESULT:** PASS FAIL (circle one) _____ initials

❑ REGRESSION
Checks only that output files from gpb_tdp_L0.pro are identical from one version of lasp-x.x to the next.

TEST
The following requires a set of VC files that contain all packet types (this can be just one file).

i. Run gpb_tdp_L0.pro in both the released and new lasp-x.x code directory with the keyword /CHECKONLY in the command line. This will generate output files p100.tmp, p200.tmp, p300.tmp, events.tmp and snaptemp.tmp.
ii. Do a UNIX binary “diff” for each type of file between released and new, e.g. “diff /apps/supported/lasp-1.7/src/tdp/p100.tmp /apps/supported/lasp-1.8/src/tdp/p100.tmp”. Do this for each .tmp file in turn.
iii. If the command above returns to the prompt with no output, that means there is no difference between the two files.

PASS/FAIL
Each comparison must yield no difference between released and new code output *.tmp files.

**RESULT:** PASS FAIL (circle one) _____ initials

## 9.4 TDP3: IPDU Time Correlation

❑ (B) ❑ (R) _____ initials

Test Case Verification Number: TDP3

INTRODUCTION
This test verifies that process_tcor.pro properly breaks out IPDU header times from VC0 files and correctly performs a least-squares fit analysis.

❑ BASELINE

APPROACH
An independent verification program is developed, TERLtcor.pl. TERLtcor.pl emulates TDP’s process_tcor (but uses a slightly different least squares algorithm). Process_tcor.pro strips IPDU headers out and stores their Earth Receipt Time (ERT) records. It uses these IPDU header ERT times for analysis against spacecraft clock times for 300 p100 time stamps and produces a least squares fit along with the raw data (for 300 points). The outputs from both programs are compared (using the same input data).

FEATURES TO BE TESTED
-- Properly parse VC0 file IPDU headers and create file(s) for import to database.
-- Properly comparing IPDU header ERT times with vehicle clock (VTCW) times for each transfer frame.
-- Correctly does least-squares analysis between ERT and VTCW times.

-- Ignores VC1and2 data, ignores any 1K or 2K format data
-- Performs same sanity checks as gpb_tdp_L0.pro for validity of data packet (such as date stamps over 2006, improper packet ID or length, et cetera)

FEATURES NOT TO BE TESTED

- Presupposes that any clock resets or time forward jumps are not in the input file (IPDUsplice.pro has been cleanly run).
- None. Please note that routine does not account for other factors involved in time correlation, such as time delays caused by: transmission from on board computer to spacecraft antenna, spacecraft data transmission from the craft to the ground, data transmission from the antenna to the front-end processor, et cetera.

TESTS
The following requires a VC0 file:

i. Process the VC0 file with process_tcor.pro.
ii. Process the VC0 file with TERLtcor.pl.
iii. Compare the TDP output file (i) with the test program output file (ii).

PASS/FAIL
Passes if all entries for both files match. All entries between tcor_raw input files to data tables must match exactly. The least-squares c0 and c1 coefficient results should match to within 0.001.

**RESULT:** PASS FAIL (circle one) ______ initials

❑ REGRESSION
Process_tcor.pro is the code that processes IPDU header data and generates a least-squares fit of vehicle time against ground receipt time. To verify that its output is the same between versions, follow the same binary UNIX diff comparison steps listed above in TDP2, but run the process_tcor routine using the keywords /CHECKONLY, /FIT, and /RAW to get the proper output files. The files to compare are tcor_raw.tmp and tcor_fit.tmp.

PASS/FAIL
Each comparison must yield no difference between released and new code output *.tmp files.

**RESULT:** PASS FAIL (circle one) ______ initials

## 9.5 TDP4: Snapshot Building

❑ (B) ❑ (R) ______ initials
Test Case Verification Number: TDP4

❑ BASELINE

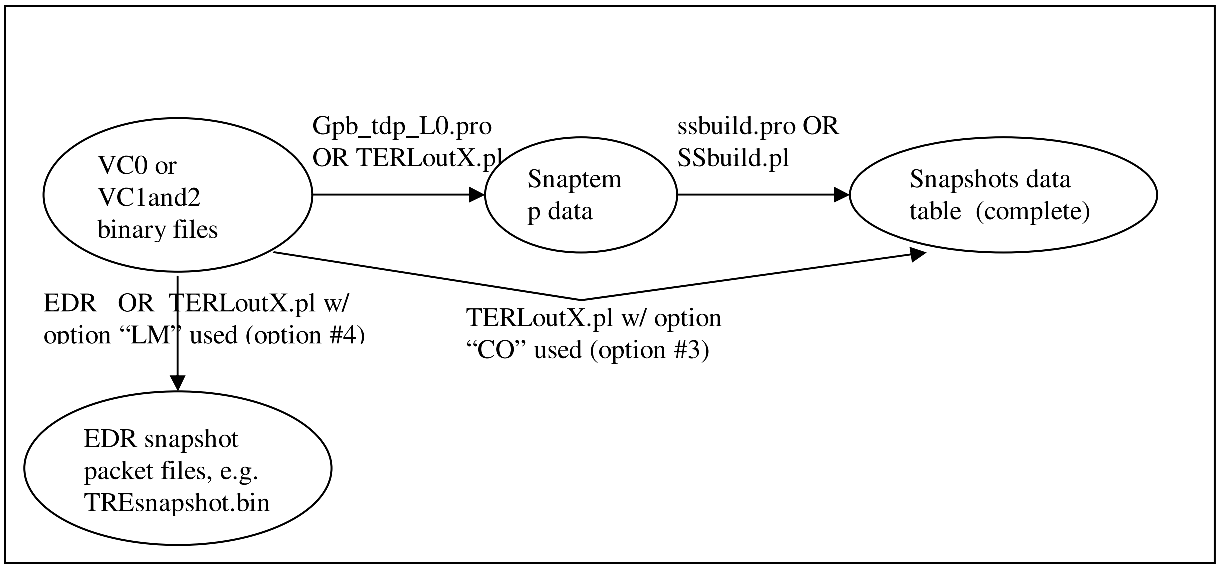
INTRODUCTION
This test verifies the accuracy of snapshot packet assembly into full snapshots. The spacecraft sends down snapshots in small packets, and depending on the type, the number of packets making up a full snapshots varies. The CSCI gpb_tdp_L0.pro processes these packets into a holding area in Sybase called GPB_L0..Snaptemp. The file that makes up Snaptemp's contents is called snaps.tmp.
Specifically, this verification checks that the content of the snaps.tmp file matches the analysis done by LM's EDR (which is a verified deliverable CSCI). Then, to verify assembly, it checks that the snapshots.tmp output file of SU's TDP ssbuild routine (to be bcp'd into Sybase as GPB_L0..Snapshots end product)

matches a third party program's full analysis. Because LM's EDR does not do full packet assembly on snapshot packets, full verification is only possible using third party analysis.

APPROACH
Four verification programs are developed.
- Use a VC1and2 file with as many types of snapshots taken as possible. Verification may require more than one file.
- Process this file through five programs: gpb_tdp_L0.pro, LM EDRS, ssbuild.pro, ssbuild.pl and TERLoutX.pl. It is possible that running code of LM EDRS will not be required after a number of runs - it does not fully process snapshots and prior testing has verified that the third party program mimics it exactly.
- Compare results and look for zero differences for analytic success.

The verification approach is more easily understood with a diagram.

**Figure 1: Snapshot Processing Options for Raw Spacecraft Binary Files**


VC0 or VC1and2 binary files
Gpb_tdp_L0.pro OR TERLoutX.pl
Snaptemp data
ssbuild.pro OR SSbuild.pl
Snapshots data table (complete)
EDR OR TERLoutX.pl w/ option "LM" used (option #4)
TERLoutX.pl w/ option "CO" used (option #3)
EDR snapshot packet files, e.g. TREsnapshot.bin

i. The TERLoutX.pl program duplicates both TDP (gpb_tdp_L0.pro) and EDR processing – i.e. one inputs a VC file, and this program outputs either a SU TDP binary file called snaps.tmp or a LM EDR binary files. Additionally, this test program can output "ssbuild.pro" format files.
ii. The ssbuild.pl program duplicates the TDP (ssbuild.pro) processing.
iii. Snapshot "exploder" is used to sort the output the 1st verification program so that it matches the output of ssbuild.pro.

Input VC file is processed five times:
i. Once by SU TDP, completing ssbuild.pro processing to final GPB_L0..Snapshots table.
ii. Once by LM EDR, if required.
iii. Once by TERLoutX.pl (to output snaps.tmp) and then by ssbuild.pl (using snaps.tmp as input) to output SU TDP files (snapshots.tmp).
iv. Once by TERLoutX.pl to output the LM EDR files.
v. Once by TERLoutX.pl to output the final "snapshots.tmp" file.

Compares are done between (i) and (iii), then between (ii) and (iv), and then between (iii) and (v).

FEATURES TO BE TESTED

- Properly assemble Snaptemp contents into complete snapshots.
- Recognize but not pass on partial snapshots to Snapshots table.
- Report on number of snapshots comprised and number of partial snapshots, and report on different types of snapshots

FEATURES NOT TO BE TESTED

- Any housekeeping tasks necessary
- Speed of assembly

TESTS
The following requires one, possibly many, VC1and2 file(s) so that all snapshot types being tested are in the sample data:

i. Process the VC1and2 file with SU TDP, completing ssbuild.pro processing to final GPB_L0..Snapshots table.
ii. Process VC1and2 file using LM EDR, if required.
iii. Process VC1and2 file by TERLoutX.pl (to output snaps.tmp) and then by ssbuild.pl (using snaps.tmp as input) to output SU TDP files (snapshots.tmp).
iv. Process VC1and2 file by TERLoutX.pl to output the LM EDR files (if required).
v. Process again by TERLoutX.pl to output the final “snapshots.tmp” file.

OVERALL PASS FAIL
Comparisons are done between (i) and (iii), then between (ii) and (iv), and then between (iii) and (v). All differences should be zero.

**RESULT:** PASS FAIL (circle one) ______ initials

❑ REGRESSION
Ssbuild is the code that processes snapshots from the Level 0 Snaptemp table to the Level 0 Snapshots table. To verify its output is the same between versions, follow the same steps as outlined above in TDP2 and compare snapshots.tmp from both versions. Any differences should be explained by code changes and specific analysis, and baseline test should be considered.

PASS/FAIL
Each comparison must yield no difference between released and new code output snapshots.tmp file.

**RESULT:** PASS FAIL (circle one) ______ initials

## 9.6 TDP6: Level 0 to Level 1 reprocessing

❑ (B) ❑ (R) ______ initials

Test Case Verification Number: TDP6

INTRODUCTION
The L0 to L1 Reprocessing tool (named l02l1) is designed to access the Level 0 Sybase database table named Packets and retrieve all APID 100 packets within the time range specified by the user for reprocessing into the Level 1 Sybase database. L02l1 is a stand-alone tool. L02l1 is a unix shell script which runs L0_to_L1.pro written in IDL with calls to the C functions in the LASP db_routines program. The user interface screen for this tool is L0_to_L1_display.pro.

❑ BASELINE

APPROACH
Run the l02l1 script for a specified time period and cycle number and compare the output file (p100.tmp) against the data in the database (GPB_L0..Packets) for the same cycle and time period. If an original p100.tmp file is available for comparison, it is preferable to use this and its corresponding cycle number and time period.

FEATURES TO BE TESTED
- Whether the l02l1 GUI functions properly
- Whether or not the correct packets are selected from the GPB_L0 database

FEATURES NOT TO BE TESTED
- Any attempted duplication of data (e.g. if that set of packets has already been reprocessed)

TESTS
i. Choose a time period and cycle number.
ii. Run l02l1 to retrieve all APID 100 packets for that time period and cycle number.
iii. Extract the APID 100 data from the database for that same time period and cycle number into a file with the same structure as p100.tmp, or use the original p100.tmp file if available.
iv. Use the UNIX "diff" tool to compare the two files. There should be no differences.
v. Check the file size for each file. These should be the same.
vi. Perform a visual inspection of the two data files using the UNIX "od" octal dump tool. There should be no differences.

PASS/FAIL
After comparing the p100.tmp file with the corresponding data in the GPB_L0..Packets table (or the original p100.tmp file), there should be no differences.

**RESULT:** PASS FAIL (circle one) _____ initials

❑ REGRESSION

TEST
Level 0 to Level 1 reprocessing suite of tools includes the l02l1 startup script, the L0_to_L1.pro processing routine and the L0_to_L1_display.pro user interface screen. If any of these three files are changed, do a binary comparison of the output file of this tool, packet100.tmp, between the old version of the code and the new version (binary diff as detailed in TDP2 above). Any differences should be explained by documentation and a baseline test should be considered.

PASS/FAIL
Each comparison must yield no difference between released and new code output packet100.tmp file.

**RESULT:** PASS FAIL (circle one) _____ initials

# 10 TEST CASES FOR LEVEL 1

Test case components are listed in the table below, along with whether or not they need to be run and if so, initials of who has run the tests. The objective for testing each component is always to get a zero difference when comparing the old revision's output files to those from the new revision. Tests are considered passed if no differences are found in running binary or ASCII differences between old files and new files; if there are differences they must be explained by accompanying documentation such as a revised or new Sdoc. Pass/Fail criteria for each test are set forth below, and supercede any general instructions for specific test cases. Test set up requires that no further modifications will be made to code and that no code is checked out of RCS for editing. Execution procedures and result descriptions follow each case below.

If the date on any of the following files has changed since the previous software release, the tests in the corresponding section number must be run and checked off for verification.

LASP version being tested: ________________________________

| **VERIFICATION REQUIRED** • **Baseline (B)** • **Regression (R)** | **FILE (under src/tdp)** | **UNIX date & filesize on program file** | **TEST NAME** | **TEST SECTION** | **VERIFIED BY** *(initials)* | **DATE** |
|---|---|---|---|---|---|---|
| | gpb_tdp_L1.pro | | TDP5 | Section 10.1 | | |
| | Post_check.pro | | TDP9 | Section 10.2 | | |
| | Process_gps.pro | | TDP7 | Section 10.3 | | |
| | TandO.pro | | TDP8 | Section 10.4 | | |
| | Average_analog.pro | | TDP12 | Section 10.5 | | |

## 10.1 TDP5: Level 1 Processing

☐ (B) ☐ (R) ______ initials

Test Case Verification Number: TDP5

☐ BASELINE

INTRODUCTION

The baseline test proves by analysis that TDP (the non-real-time Telemetry Data Processing software) is correctly processing data from the binary contents of the Level 0 database, which originate from raw, Gravity Probe B spacecraft binary files (please reference section 9 of this document for verification of Level 0 processing). The analysis includes the correct correlation of spacecraft vehicle time to each mnemonic.

APPROACH

One outside verification program, TERL_L1, is developed to emulate the end results expected by the Level 1 database.

Use a VC1and2 file, a VC0 file or a framex file (any spacecraft file is acceptable) and process through gpb_tdp_L0.pro and then gpb_tdp_L1.pro. Process this same file using TERL_L1 and compare the outputs of both processes. These outputs should be identical.

FEATURES TO BE TESTED

- Accuracy of Level 1 data processing on APID 100 Packets
- Spacecraft time stamping for each mnemonic

- Unpacking of parent to child mnemonic data
- Decommutation
- MSS version-dependent decommutation

FEATURES NOT TO BE TESTED

- Speed of processing
- Any packet processing that is not APID 100
- Formation of derived monitors

TESTS

The test requires a clean (sent through IPDUsplice.pro) spacecraft file in the form of a VC1and2 file, a VC0 file or a framex file from a test bed system.

i. The spacecraft file must be processed by TDP (using gpb_tdp_L0.pro) to create Level 0 data, thus creating an output file named p100.tmp, which is full of APID 100 packet data.
ii. This same file, p100.tmp, must be run through both systems – TERL_L1 and TDP – generating independent results in two different parts: analog and discrete.
iii. Perform the following steps are performed on the analog and discrete data generated by both TERL_L1 and TDP:
    a. Select the first 50,000 frames from each file for comparison
    b. Pipe all outputs from TERL_L1 and TDP to ascii files.
    c. Perform the UNIX "sdiff –s" function on the TERL_L1 analog data vs. the TDP analog data, and again on the TERL_L1 discrete data vs. the TDP discrete data.
iv. Perform the steps listed above for the /scionly or /engonly data (1K and/or 2K data is considered "/engonly", 32K data is considered "/scionly). In TDP, the 1K/2K data will go to two different ascii files than the two files created for 32K data, so two repetitions may be required if the source data has mixed telemetry rates.

PASS/FAIL
After comparing TDP discrete to TERL_L1 discrete, and TDP analog to TERL_L1 analog, the differences should be zero with the following exceptions:

- TERL_L1 processes its zeroes as "0" and TDP processes zeros as "-0" and "0". TERL_L1 processes all "0" and "-0" as "0" while TDP keeps "-0" and "0" as is from the spacecraft.

**RESULT:** PASS FAIL (circle one) ______ initials

❑ REGRESSION

TEST
Choose a p100.tmp or packets100.tmp data file (generated by gpb_tdp_L0.pro or the l02l1 tool – this file can be from the same testing run used in section 9 above) preferably containing as many different formats as possible to enable a larger comparison.

i. Run gpb_tdp_L1 with /checkonly keyword on the p*.tmp file file. See P0826 for further instructions on processing Level 1 data if necessary. Note that the p100 file (or packets100 file) should be processed using the same MSSID).
ii. Do the UNIX command "diff" on each binary file (listed below) in the released version of lasp against test version of lasp. (e.g. 'diff filenameA filenameB' where filenameA and filenameB are the output files from the old and new versions of gpb_tdp_L1.pro, respectively).
    a. Testing analog data types: compare tmanalog.tmp old vs new, compare snanalog.tmp old vs new (if 1K/2K data is in source file).
    b. Testing discrete data types: compare tmdiscrete.tmp old vs new, compare sndiscrete.tmp old vs new (if 1K/2K data is in source file).
    c. Testing general functionality of input to limit checking routine: compare tmminmax.tmp and tmbadstate.tmp old vs. new.

Any differences should be explained by changes made to code. Documentation must be produced to explain any differences made by code (such as a revision of S0613). If there were no gpb_tdp_L1, gpb_db.pro or dbroutines.so code changes, there should be no differences.

PASS/FAIL
Each comparison must yield no difference between released and new code output files.

**RESULT:** PASS FAIL (circle one) _____ initials

## 10.2 TDP9: Post Check

❑ (B) ❑ (R) _____ initials

Test Case Verification Number: TDP9

INTRODUCTION
This test case is used to verify the generation of limit checking and state checking summaries, which are generated after the Level 1 data processing.

❑ BASELINE

APPROACH
Comparison of generated file against database selections and TCAD plots using post_check.pro.

FEATURES TO BE TESTED
- Reporting analog data samples which exceed the limit settings in the MSS database.
- Reporting discrete values which are outside the acceptable states for this variable.
- Calculation speed.

FEATURES NOT TO BE TESTED
- Validation of limit values.
- Choice of MSS database for limit generation.
- Interpretation of state and limit report by users.
- Import of state and limit report into TQSM application.

TESTS
- Record the reported time of post_check start, and the reported time of post_check completion.

  START TIME:___________________________ STOP TIME:_______________________

- After running a full level 1 import (run the gpb_tdp_L1.pro program with the /checkonly keyword if the data was previously BCP'd to the Level 1 databases) verify that the file 'postcheck.tmp' was generated during the run of this import.
  - Open the postcheck.tmp file in a text editor or "more" the file. Identify the two section headers in the file, 'Limit Checking' and 'State Checking'.
  - Choose two or more mnemonics from different subsystems that are reported as out of limits. Using the earliest violation time and the latest violation time, select a time range for this cycle in TCAD and plot the selected mnemonics with 'Display Red Limits' selected in the 'Plot Options' screen. The plotted mnemonic should be seen to cross the limit line as reported in the postcheck.tmp file.
- For secondary validation of limit checking:
  - Look up the TMID of each mnemonic, using either the TM INFO screen in TCAD or a manual selection from the GPB_L1.. TMnames data table.

  - Select * from TMlimits where MSSID=(select max(MSSID) from GPB_L1..TMlimits) AND TMID=#your_TMID#.
  - Select * from TManalog where SCT_Cycle=#your_CYCLE# and MSSID=#your_MSSID# and (Value > #Yellow_Low_Value# OR Value < #Yellow_High_Value#)
  - This will output all values exceeding limits for this mnemonic (using the highest MSSID) and may manually be compared against the values reported in postcheck.tmp
- Choose two or more mnemonics from different subsystems that have reported state violations. Using the earliest reported state violation time and the latest reported violation time, select a time range for this cycle in TCAD and plot the selected mnemonics to the screen. The plotted mnemonic should be seen to exceed the states listed in the plot.

PASS/FAIL
Post Check report is generated in under 5 minutes per 100M of data in the p100.tmp file.
Limit violations are reported accurately.
State violations are reported accurately.

**RESULT:** PASS FAIL (circle one) ______ initials

❑ REGRESSION

TEST

i. Output file from post_check.pro is called post_check.tmp. Run post_check.pro to generate output files from a tmminmax.tmp file and tmbadstate.tmp file.
ii. Do the UNIX command "diff" on each binary file in the released version of lasp against test version of lasp. (e.g. 'diff filenameA filenameB' where filenameA and filenameB are the output files from the old and new versions of process_gps.pro, respectively). Binary diff should show no differences between the files.

Any differences should be explained by changes made to code. Documentation must be produced to explain any differences made by code (such as a revision of S0613). If there were no post_check.pro, gpb_db.pro or dbroutines.so code changes, there should be no differences.

PASS/FAIL
Each comparison must yield no difference between released and new code output files.

**RESULT:** PASS FAIL (circle one) ______ initials

## 10.3 TDP7: Process GPS

❑ (B) ❑ (R) ______ initials
Test Case Verification Number: TDP7

❑ BASELINE

INTRODUCTION
The baseline test proves by analysis that all GPS data is being properly processed by TDP from the binary contents of the Level 0 database, which originate from raw, Gravity Probe B spacecraft binary files, to Level 1 data.

APPROACH
Use an independent verification program, TERLgps.pl, which reads telemetry packets (APID 100 type) from telemetry where format=155, 156, 215, 216 and 217 and interprets the contents of the GPS-specific bytes.

Query the Sybase Level 0 Packets table, by Format ID, to generate a file of p100.tmp format.
Process this file using TERLgps.pl and compare it to its corresponding Level 1 data that has already been processed using process_gps.pro (TDP).

FEATURES TO BE TESTED
- Accuracy of special GPS processing routines
- Data in formats 155, 156, 215, 216 and 217

FEATURES NOT TO BE TESTED
- Data in any other telemetry formats
- Speed of GPS processing routines

TESTS
i. Testing starts with a p100.tmp file that is created by querying the existing Sybase Level 0 Packets table (using a “where” statement on Format_ID), or has been produced in a previous test, such as in the section above. This p100.tmp file must have data in one of the formats listed above or the verification will be invalid.
ii. The p100.tmp file is processed using the independent verification program TERLgps.pl and then compared against the data found in the Sybase Level 1 database in the GPS packets and telemetry tables for the same spacecraft times.
iii. There may be existing data in the Sybase Level 1 database already processed by TDP using the process_gps.pro program. If not, run process_gps.pro with the keyword /CHECKONLY to generate the GPS table data from the p100.tmp file. Output files from process_gps.pro are named gps_pkt.tmp and gps_blk.tmp.
iv. The data from TERLgps.pl is exported to ASCII files and the existing Sybase Level 1 data (or that generated from process_gps.pro in /CHECKONLY mode) is exported to corresponding files.
v. Each set of ASCII files is compared using UNIX “diff” commands.

PASS/FAIL
The diff commands should yield null results, corresponding to no differences in TDP processing (process_gps.pro) vs. TERLgps.pl processing.

**RESULT:** PASS FAIL (circle one) _____ initials

❑ REGRESSION

TEST

i. Output files from process_gps.pro are gps_pkt.tmp and gps_blk.tmp. Use keyword /CHECKONLY to generate output files from a p100.tmp file (or packets100.tmp file).
ii. Do the UNIX command “diff” on each binary file in the released version of lasp against test version of lasp. (e.g. ‘diff filenameA filenameB’ where filenameA and filenameB are the output files from the old and new versions of process_gps.pro, respectively). Binary diff should show no differences between the files

Any differences should be explained by changes made to code. Documentation must be produced to explain any differences made by code (such as a revision of S0613). If there were no process_gps.pro, gpb_db.pro or dbroutines.so code changes, there should be no differences.

Please note that an MSS database change may affect this program. Code on this program should be checked on change of MSS versions if specific monitors are changed in MSS, per the MSS import process document, P0908. See VDD and code for more details.

PASS/FAIL
Each comparison must yield no difference between released and new code output files.

**RESULT:** PASS FAIL (circle one) ______ initials

## 10.4 TDP8: TandO, or "Timing and Orbit"

❑ (B) ❑ (R) ______ initials

Test Case Verification Number: TDP8

INTRODUCTION

For MSS delivery 3.3.3 and higher, the TandO.pro routine decommutates the first 81 bytes of each 32K telemetry format and places specific timing and GPS monitors in two separate tables. This routine creates two files (gps_timing.tmp, gps_orbit.tmp), whose contents are in turn stored in two data tables: GPB_L1..GPS_Timing and GPB_L1..GPS_Orbit. The first part of TandO handles timing mnemonics, the second part handles orbit mnemonics.

APPROACH
Simple comparison of the values of specific mnemonics in two separate data files over the same five-minute time interval.

FEATURES TO BE TESTED
- Accuracy of parsing done by tando.pro against the accuracy of gpb_tdp_L1.pro

FEATURES NOT TO BE TESTED
- Speed of tando.pro
- Fetching of any data outside the first 81 bytes of any 32K telemetry format.

TESTS
Over the same five-minute interval select the following data from the specified Sybase data tables:

i. Over the same five-minute interval select the following data into tempdb tables from the specified Sybase data tables:
   a. SF_Frame_Count, SQ_SciVehTime32, SQ_SciVehTime8, SQ_Sci10HzTime, SQ_PPSv16F_Time, SCT_Cycle and SCT_VTCW from GPB_L1..GPS_Timing.
   b. SF_Frame_Count, SQ_SciVehTime32, SQ_SciVehTime8, SQ_Sci10HzTime, SQ_PPSv16F_Time, SCT_Cycle and SCT_VTCW from GPB_L1..TManalog.
   c. SF_Frame_Count, SP_RecvrMode1, SP_GPS_WeekNum1, SP_GPSTimeWeek1, RP_ECEFPosX, RP_ECEFPosY, RP_ECEFPosZ, RP_ECEFVelX, RP_ECEFVelY, RP_ECEFVelZ, SP_GPS_Digit_B, SP_GPS_Fract, SP_Time_Trans from GPB_L1..GPS_Orbit.
   d. SF_Frame_Count, SP_RecvrMode1, SP_GPS_WeekNum1, SP_GPSTimeWeek1, RP_ECEFPosX, RP_ECEFPosY, RP_ECEFPosZ, RP_ECEFVelX, RP_ECEFVelY, RP_ECEFVelZ, SP_GPS_Digit_B, SP_GPS_Fract, SP_Time_Trans from GPB_L1..TManalog.

ii. Using the Sybase bcp utility, bcp out the four tempdb tables created in step (i). Do an ASCII diff and visual inspection on the data in the exported tables created in steps (a) & (b) and then against those created in steps (c) & (d).

PASS/FAIL

Visual inspection is required because the tando.pro data is stored in its spacecraft natural and most dense form (4 byte floats and 2 and 4 byte long integers). Therefore, it will never directly nor binarily agree with TDP Level 1 data that is stored in 8 byte floats.

Accordingly, TIMINGget and ORBITget (.pl) retrieve and convert the data into human engineering values to be compared. Plus, tando.pro filters out a bunch of imperfect "orbit" conditions by SF_MSS_ID, by non science 32k data, by SF_Frame_Count, and by byte location (first 81 bytes), in conjunction with assembling only data from same orbit solution, and by tossing data of RecvrMode1 not x50 (80 decimal).

There should be some differences with respect to the "T" part of tando.pro (the Timing output in steps a & b) and Level 1 analog data, as that tando.pro only extracts data from: the first 81 bytes (555 format), the 32k science formats, of SF_MSS_ID >= 3.3.3.

There should be definite differences in the "O" part of tando.pro (the Orbit output in steps c & d) with respect to Level 1 analog data.

There should be no differences in the comparison discussed in step (ii).

**RESULT:** PASS FAIL (circle one) _____ initials

❑ REGRESSION

FEATURES TO BE TESTED

- Consistency of data created by tando.pro between one code release and the next

FEATURES NOT TO BE TESTED

- Correctness of tando.pro
- Time it takes to perform tando.pro.

TEST

i. Output files from tando.pro are gps_timing.tmp and gps_orbit.tmp. Run tando.pro with the /CHECKONLY keyword to generate output files. Do this in the directories of the released version of the software and the new version of the software.
ii. Do the UNIX command “diff” on each binary file in the released version of lasp against test version of lasp. (e.g. ‘diff filenameA filenameB’ where filenameA and filenameB are the output files from the old and new versions of process_gps.pro, respectively). Binary diff should show no differences between the files.

Any differences should be explained by changes made to code. Documentation must be produced to explain any differences made by code (such as a revision of S0613). If there were no significant MSS changes to the first 81 bytes of every 32K format, to tando.pro, gpb_db.pro or dbroutines.so code changes, there should be no differences.

PASS/FAIL
Each comparison must yield no difference between released and new code output files.

**RESULT:** PASS FAIL (circle one) _____ initials

## 10.5 TDP12: Average Level 1 TManalog data

❑ (B) ❑ (R) _____ initials

Test Case Verification Number: TDP12

INTRODUCTION
In order to control the size of the telemetry data in the Level1 database, the TManalog table is averaged on five-minute centers. The routine used to run this process is average_analog.pro, and the averaged data is saved in the TMaverage table. To decide which mnemonics are to be averaged, the routine reads TMavgid data table for real, analog values from 32K format data.

Full-resolution (all of the analog data) is all data for monitors sampled at the spacecraft rate of every 10th of a second. The averaging of TManalog data is baselined for every few weeks; however, the frequency of the averaging process can be changed by the database administrator. The content of the averaged data consists of the minimum, maximum, average and standard deviation values (per five-minute interval) for each monitor sampled over that five-minute interval, and the number of times the monitor was sampled during the interval.

According to requirement MOC-017, the data processing shall be able to retrieve and display averaged programmable telemetry data for a selected monitor from the entire mission set within 1 hour of a request.

☐ BASELINE

APPROACH
To test speed, run TCAD, choose a monitor and measure the amount of time it takes to show data. Do this with several different high-frequency sampled monitors for performance measurements. To show capability of viewing averaged data, run TCAD, choose a monitor and select a time interval and cycle number that retrieves averaged data.

To test accuracy and correctness of average_analog.pro, select a monitor from TCAD with the “Averaged” box checked in TCAD’s “setup” options. After plotting data, zoom in on any segment of data and check the accuracy of the given numbers via visual inspection of the plot. Alternatively, if zooming does not provide an adequate inspection, get the data via table and perform a numerical inspection if desired.

FEATURES TO BE TESTED
- Time it takes to retrieve averaged data for display from the TMaverage table.
- Accuracy of averaged data w.r.t. minimum, maximum and averaged points.
- Correctness of average_analog.pro

FEATURES NOT TO BE TESTED
- Deletion of any data from the TManalog table (which is not done by average_analog.pro)
- Averaging SNanalog data (no 1K or 2K data is averaged by this routine)

TESTS
i. First, check the amount of data in the average data table, GPB_L1..TMaverage via a select count(TMID) statement in Sybase. If there is no data, create some using P0905, “Processing Data from High-resolution to an Averaged Data Set”.
ii. Start TCAD and chose monitors to display. The choice of which monitor(s) to display may be influenced by the existence of those TMIDs in the TMaverage data table. That is, one should choose monitors that will return data instead of an empty set.
iii. Time the arrival of results.
START TIME ________________________ END TIME ____________
iv. Inspect the results of the averaged data against the analog data. Does the inspection yield a satisfactory averaging? ______Yes ______No

PASS/FAIL

Successfully retrieve and display averaged programmable telemetry data for a selected monitor from the entire mission set within one (1) hour. Averaged values of data passed visual inspection for plot or table values.

**RESULT:** PASS FAIL (circle one) _____ initials

❑ REGRESSION

FEATURES TO BE TESTED

- Consistency of data created by average_analog.pro between one code release and the next

FEATURES NOT TO BE TESTED

- Correctness of average_analog.pro
- Time it takes to retrieve averaged data for display from the TMaverage table.

TEST

iii. Output file from average_analog.pro is called tmaverage.tmp. Run average_analog.pro with the /CHECKONLY keyword to generate an output file. Do this in the directories of the released version of the software and the new version of the software.

iv. Do the UNIX command "diff" on each binary file in the released version of lasp against test version of lasp. (e.g. 'diff filenameA filenameB' where filenameA and filenameB are the output files from the old and new versions of process_gps.pro, respectively). Binary diff should show no differences between the files.

Any differences should be explained by changes made to code. Documentation must be produced to explain any differences made by code (such as a revision of S0613). If there were no average_analog.pro, gpb_db.pro or dbroutines.so code changes, there should be no differences.

PASS/FAIL
Each comparison must yield no difference between released and new code output files.

**RESULT:** PASS FAIL (circle one) _____ initials

## 11 TEST CASES FOR CORE PROGRAMS

| **VERIFICATION REQUIRED** • **Baseline (B)** • **Regression (R)** | **FILE (under src/tdp)** | **UNIX date & filesize on program file** | **TEST NAME** | **TEST SECTION** | **VERIFIED BY** *(initials)* | **DATE** |
|---|---|---|---|---|---|---|
| | gpb_db.pro | | CORE1 | Section 11.1 | | |
| | gpb_time.pro | | CORE1 | Section 11.1 | | |
| | Generic_time.pro | | CORE1 | Section 11.1 | | |
| | db_routines.so | | CORE1 | Section 11.1 | | |

### 11.1 CORE1: dbroutines.so, gpb_db.pro, generic_time and gpb_time.pro

Underlying code layers include files named dbroutines.so, gpb_db.pro, generic_time.pro and gpb_time.pro. These are underlying software layers between all tdp and tcad routines. If any of these files have changed, do regression testing for TDP2, TDP5, TDP7, TDP8, TDP9 and TDP12, along with TCAD regression tests listed in separate document.

**REQUIRED?**

Regression of TDP2 TDP5 TDP7 TDP8 TDP9 TDP12 and TCAD P0950

YES NO (circle one) _____ initials

## 12 GLOSSARY

This section contains an alphabetic list and definitions of all acronyms used in the document, all proper nouns, and any words used in a non-standard way.

| Word | Detail |
|---|---|
| CSCI | Computer Software Configuration Item |
| LASP | Laboratory for Atmospheric and Space Physics, University of Colorado |
| moc-server | Host name of the SUN computer that is the primary server for the MOC. |
| Science server | Host name of the SUN computer which is the primary server for science LAN |
| SAFS | Standard Autonomous File Server (GSFC facility) |
| TCAD | Telemetry Checking, Analysis, and Display |
| TDP | Telemetry Data Processing |

---

Successful Completion of P0949:

RUN BY (signature)___________________________________ Date: ______________

Print Name: __________________________________________

ALL TESTING PASSED (CIRCLE ONE): YES – PASSED NO – MCR FILED

QA Review of process _________________________________ Date: ______________