VSIPL 1.0 API Test Suite

(Core Lite Profile)

Dan P. Campbell
Georgia Tech Research Institute

Randall S. Janka
Georgia Tech Research Institute

http://www.vsipl.org

August 18, 2000

This material is based in part upon work supported by Ft. Huachuca/DARPA under contract No. DABT63-96-C-0060. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of Ft. Huachuca/DARPA.

1. Introduction

1.1 Scope

This document describes the API (application programming interface) compliance test suite for the Vector Scalar Image Processing Library (VSIPL) and assists a user in building, running, and using the test suite. It is limited to what is known as the "Core Lite Profile," which is the fundamental subset of the complete v1.0 VSIPL specificiation. The VSIPL API has been developed by a consortium made up of leading industry vendors, academic institutions, interested users, and government laboratories. The Georgia Tech Research Institute (GTRI) was funded by the US Navy as a neutral third party to develop a test suite to ensure that shareware and commercial implementations comply to the VSIPL specification.

1.2 Definition of Compliance

Strictly speaking, to comply means "to act in accordance with prevailing standards or customs." Ultimately, determining whether or not a VSIPL implementation complies with the API specification requires comparing its functionality to what has been delineated in the VSIPL API specification document. However, to provide a fair and reasonable measure of compliance, the test suite has been created. Passing the VSIPL test suite is a necessary but not a sufficient condition to ascertain whether an implementation is in compliance. It merely provides a bar that is high enough to assure a reasonable measure of compliance.

1.3 Supporting Documentation

Documentation describing the VSIPL API, profiles, implementation requirements, and other complementary documents are all freely available on the VSIPL website at http://www.vsipl.org.

1.4 Copyright Language

The language describing the copyright of the VSIPL test suite is based on the Free Software Foundation's General Public License (GPL) and is included in Appendix A of this document.

2. Installation

2.1 Unpacking

Once downloaded, the test suite may be unpacked by issuing the following commands on most systems:

```
gzip -d TestSuite.tar.gz

tar xfv TestSuite.tar
```

These commands will create a directory called

```
TestSuite
```

If you are unable to unpack the files please email testsuite@vsipl.org for assistance.

2.2 Configuration

Under most standard systems, only the first section of the Makefile is of note – the macros in the makefile must be set to match the location of the VSIPL librbary, and any adjustments to the standard compile command lines must be made here. However, there are several special issues to consider:

2.2.1 Macro Configuration

The top level makefile contains several macro definitions that you may wish to modify for your system.

LIB_VSIP: This is the filename of the VSIPL library that will be used in the linking stage.

LIB_VSIP_LOC: This must be set to the full path of the location of the VSIPL library to be linked for the Test Suite.

VSIP_INC: This must be set to the full path of the `vsip.h` header file for the implementation of VSIPL that is to be tested.

CC: This macro must be set to the name of the compiler that is to be used to build the Test Suite.

CFLAGS: Any additional flags that must be sent to the compiler should be defined here.

DEFINES: Any additional `#define's` to be sent to the compiler should be defined here.

HEADERS: Any additional headers that must be included in compiles must be defined here.

LD: This macro must be set to the name of the linker that is to be used to build the Test Suite.

LFLAGS: Any additional flags that must be sent to the linker should be defined here.

EXTRALIBS: Any additional libraries required by the target platform should be defined here.

2.2.2 Text Output

The stock Test Suite sends all output both to `STDERR`, and to the file "`vsiptest.out`". The Test Suite, however, is intended to run on systems of varying levels of output capabilities. If you wish to change the stock output method, you will need to modify the function `vtprintf()` found in the `util/vsiptest_io.c` source file.

2.2.3 Malloc

The VSIPL API requires certain behavior when blocks or views can not be created. In order to test compliance with this requirement, the test suite assumes that all VSIPL object creation will make use of `malloc()` at some point. While this is obviously not universally true, it is the closest to universal way to control success and failure of object creation. Making use of this assumption, the test suite ships with a custom modification of GNU's `malloc()` utility. If the VSIPL implementation that is to be tested does not use `malloc()`, or if there are difficulties compiling or linking the malloc section of the Test Suite, it may be removed by changing the following line in the top level make file:

```
LFLAGS=-L$(LIB_VSIP_LOC) -lm ../malloc/alloc.o
```

to

```
LFLAGS=-L$(LIB_VSIP_LOC) -lm
```

And removing the following line:

```
(cd malloc; make alloc.o CC='$(CC)' CFLAGS='$(CFLAGS)')
```

2.3 Building

Once the configuration is complete, the Test Suite may be built by issuing the command `make` from the top level directory:

3. Using the Test Suite

The Test Suite is made up of a collection of individual function tests. Each VSIPL function required by the VSIPL Core Lite Profile, with the exceton of support functions for the signal processing functions, has an individual function test associated with it within the Test Suite. The Test Suite has three methods of compliance testing. The first is the compile and link phase. In order to meet the Core Lite Profile, an implementation must contain all of the functions and define all of the macros required by the Core Lite Profile. If an implementation is missing any of these, the Test Suite will not build properly. The second is explicit testing – functions that are required to behave in a certain way are executed, and their behavior is observed. If any behavior is not as expected, an error is generated, and the test fails. The final method is implicit side effect testing, which is not done directly by the Test Suite. Certain issues, such as memory management, are not directly testable within the test suite. For certain fucntions (e.g. `vsip_blockdestroy_f`) there are potential system level side effects, and the Test Suite attempts to make any side effects observable externally by iterating the function a large number of times. An external suite of memory leak detection tools would make an excellent complement to the Test Suite.

The tests within the Test Suite are organized into groups called modules. Each module is a set of tests on VSIPL functions which are similar in nature. The modules included in the Test Suite are "block", "view", "elwise", and "sigproc". Each module, and the tests within it, is described below.

The normal mode of operation of the Test Suite is to execute all tests and to report a "pass" or "fail" for the VSIPL implementation being tested. Alternatively, individual modules may be launched. This will allow the compliance testing of only a specified subset of the VSIPL implementation. Finally, each test can be run individually, with an exit code of 0 indicating a passing function, or 1 for a function which does not pass all of the tests.

3.1 Launching the Test Suite

The Test Suite may be run simply by typing the following commands, from the top level directory:

```
cd EXEC

vsiptest
```

These commands will run the full suite of tests with the default level of verbosity.

The Test Suite, in normal operation, will iterate through each of the groups of functions (block, view, elwise, and sigproc) covered by the Core Lite profile and run a series of tests relevant to each function. The group of functions currently being tested

will be displayed, as will the specific function being tested, with a "pass" or "fail" result. A pass or fail result will be listed for each group of functions, as well as for the whole suite of functions. Any test failing within any sub-group will generate a "fail" for that sub-group.

3.2 Options

3.2.1 Executing Individual Modules and Function Tests

Individual groups of functions may be tested using one of the following commands:

```
vsiptest_block

vsiptest_view

vsiptest_elwise

vsiptest_sigproc
```

The full list of functions tested by each sub-group is given below.

Individual VSIPL functions may be tested by changing into the directory of the module in which the function resides, and executing `vsiptest_`*`functionname`*. For example, the commands:

```
cd block

vsiptest_blockcreate_f
```

will execute the tests on `vsip_blockcreate_f`.

3.2.2 Numerical Accuracy Reporting

The Test Suite contains several levels of numerical accuracy reporting for each of the functions within the `sigproc` and `elwise` modules. Each of these functions is executed, and the results are compared to data generated by Matlab. The intent of the Test Suite is to test functionality rather than a particular level of accuracy, so the Test Suite only tests accuracy within a very broad range, which may be changed by implementers. This is intended to allow the Test Suite to verify that functions behave as described by the VSIPL API specification without imposing an arbitrary accuracy requirement. However, in order to allow users the ability to gauge the accuracy of implementations of VSIPL, the Test Suite reports some basic accuracy reporting. In order to enable accuracy reporting, a command line argument of `1` or `2` may be supplied to any test. For example

```
vsiptest 1
```

will launch the full suite of tests with accuracry reporting of level one on.

Level one reporting will give an overall summary for each function test of the sub-test with the largest observed error. The test suite will report the sub-test with the maximum error, the maximum error observed, and the root mean square of the errors observed for all data within that subtest.

Level two reporting will give an error report for each subtest within a given function test. The function test will report the name of each subtest, with the maximum error observed for that subtest, as well as the root mean square of the observed errors for all data within that subtest.

Individual function tests run with accuracy reporting will report to `stderr` as well as to the file `vsiptest.out`. Test Suite `vsiptest_` scripts launched with accuracy reporting will not display accuracy results to `stderr`.

Finally, the tolerance level used by the Test Suite for each function with numerical testing may be shown by executing:

```
vsiptest tolerances
```

from the top level directory. This will display the threshold error tolerance used for each function test.

4. Modules

4.1 Block – VSIPL Block Support Functions

The “block” module contains tests for the VSIPL functions that operate directly on or with blocks, as well as the `vsip_init` and `vsip_finalize` functions. Since these functions are support functions and do not have any observable effects directly, the tests generally consist of using the tested function in as many diffferent ways as is feasible. Then other VSIPL functions with observable effects are performed in an attempt to determine whether all modes of a block function operate properly. The tests for each function are detailed below

4.1.1 vsiptest_blockadmit_f

4.1.1.1 Attempt to admit a simple block with update=1

4.1.1.2 Compare elements block to expected values

4.1.1.3 Attempt to readmit block

4.1.1.4 Attempt to admit invalid (destroyed) block

4.1.1.5 Attempt to admit invlaid (bound to NULL) block

4.1.2 vsiptest_blockadmit_l

4.1.2.1 Attempt to admit a simple block with update=1

4.1.2.2 Compare elements block to expected values

4.1.2.3 Attempt to readmit block

4.1.2.4 Attempt to admit invalid (destroyed) block

4.1.2.5 Attempt to admit invlaid (bound to NULL) block

4.1.2.6 Attempt to admit VSIP created block

4.1.3 vsiptest_blockbind_f

4.1.3.1 Attempt to bind a block to NULL

4.1.3.2 Compare first element in bound view to expected value

4.1.3.3 Compare last element in bound view to expected value

4.1.3.4 Compare length of bound view to expected value

4.1.4 vsiptest_blockbind_l

4.1.4.1 Attempt to bind a block to NULL

4.1.4.2 Compare first element in bound view to expected value

4.1.4.3 Compare last element in bound view to expected value

4.1.4.4 Compare length of bound view to expected value

4.1.5 vsiptest_blockcreate_f

4.1.5.1 Compare first element to value assigned

4.1.5.2 Compare last element to value assigned

4.1.5.3 Compare size of block to expected value

4.1.6 vsiptest_blockcreate_l

4.1.6.1 Compare first element to value assigned

4.1.6.2 Compare last element to value assigned

4.1.6.3 Compare size of block to expected value

4.1.7 vsiptest_blockdestroy_f

No output – this test creates and destroys a block iteratively many times.

4.1.8 vsiptest_blockdestroy_l

No output – this test creates and destroys a block iteratively many times.

4.1.9 vsiptest_blockfind_f

Bind a block and fill it with data.

4.1.9.1 Verify that blockfind returns non-NULL

4.1.9.2 Compare data at pointer returned to original block

4.1.9.3 Verify that blockfind on VSIPL block returns NULL

4.1.10 vsiptest_blockfind_l

Bind a block and fill it with data.

4.1.10.1 Verify that blockfind returns non-NULL

4.1.10.2 Compare data at pointer returned to original block

4.1.10.3 Verify that blockfind on VSIPL block returns NULL

4.1.11 vsiptest_blockrebind_f

4.1.11.1 Attempt to admit a user block

4.1.11.2 Verify that rebind of admitted block returns NULL

4.1.11.3 Attempt to release user block

4.1.11.4 Verify that rebind to NULL returns NULL

4.1.11.5 Verify that rebind to valid pointer returns old data address

4.1.11.6 Verify that rebind of a derived block returns NULL

4.1.12 vsiptest_blockrebind_l

4.1.12.1 Attempt to admit a user block

4.1.12.2 Verify that rebind of admitted block returns NULL

4.1.12.3 Attempt to release user block

4.1.12.4 Verify that rebind to NULL returns NULL

4.1.12.5 Verify that rebind to valid pointer returns old data address

4.1.13 vsiptest_blockrelease_f

4.1.13.1 Verify that releasing a user block returns data pointer

4.1.13.2 Verify that releasing an already released block returns non-NULL

4.1.13.3 Compare elements in released block to expected values

4.1.13.4 Verify that attepmpting to release a VSIP block fails

4.1.13.5 Verify that attempting to release a derived block fails

4.1.14 vsiptest_blockrelease_l

4.1.14.1 Verify that releasing a user block returns data pointer

4.1.14.2 Verify that releasing a released block returns non-NULL

4.1.14.3 Compare elements in released block to expected values

4.1.14.4 Verify that attepmpting to release a VSIP block fail

4.1.15 vsiptest_cblockadmit_f

4.1.15.1 Attempt to bind complex block

4.1.15.2 Attempt to admit a split complex block

4.1.15.3 Compare data elements in admitted split complex block to expected values

4.1.15.4 Attempt to create an interleaved block

4.1.15.5 Attempt to admit interleaved block

4.1.15.6 Compare data elements in admitted interleaved complex block to expected values

4.1.15.7 Attempt to admit an admitted block

4.1.15.8 Attempt to admit a block bound to NULL data

4.1.15.9 Attempt to admit a VSIP block

4.1.16 vsiptest_cblockbind_f

4.1.16.1 Attempt to bind a complex block to two NULL pointers

4.1.16.2 Attempt to bind an interleaved complex block

4.1.16.3 Attempt to bind a split complex block

4.1.16.4 Verify first data element in derived imaginary block from split complex block

4.1.16.5 Verify last data element in derived imaginary block from split complex block

4.1.16.6 Verify first data element in derived real block from split complex block

4.1.16.7 Verify last data element in derived real block from split complex block

4.1.16.8 Verify first data element in derived imaginary block from interleaved complex block

4.1.16.9 Verify last data element in derived imaginary block from interleaevd complex block

4.1.16.10 Verify first data element in derived real block from interleaved complex block

4.1.16.11 Verify last data element in derived real block from interleaved complex block

4.1.16.12 Verify the size of an admitted block

4.1.17 vsiptest_cblockcreate_f

4.1.17.1 Verify the first data element assigned to a complex created block, real view

4.1.17.2 Verify the last data element assigned to a complex created block, real view

4.1.17.3 Verify the first data element assigned to a complex created block,

imaginary view

4.1.17.4 Verify the last data element assigned to a complex created block, imaginary view

4.1.17.5 Verify the size of a created block

4.1.18 vsiptest_cblockdestroy_f

No output – this test creates and destroys a block iteratively many times.

4.1.19 vsiptest_cblockfind_f

4.1.19.1 Verify split block returned values

4.1.19.2 Verify split block returned value array

4.1.19.3 Verify second pointer returned from interleaved block is NULL

4.1.19.4 Verify first pointer returned from interleaved block is non-NULL

4.1.19.5 Verify interleaved block returned value array

4.1.19.6 Verify that function called on VSIP block fails

4.1.20 vsiptest_cblockrebind_f

4.1.20.1 Attempt to admit interleaved block

4.1.20.2 Attempt to admit split block

4.1.20.3 Attempt to rebind admitted split block

4.1.20.4 Attempt to rebind admitted interleaved block

4.1.20.5 Attempt to release split block

4.1.20.6 Attempt to release interleaved block

4.1.20.7 Attempt to rebind split block to NULL values

4.1.20.8 Attempt to rebind interleaved block to NULL values

4.1.20.9 Verify rebind of interleaved block returns pointers to original data

4.1.20.10 Verify rebind of split block returns pointer to original data for real part

4.1.20.11 Verify rebind of split block returns pointer to original data for imaginary

4.1.20.12 Verify rebind of split block to interleaved returns original data pointers

4.1.20.13 Verify rebind of interleaved block to splti returns original data pointer

4.1.20.14 Verify that split block rebound to interleaved maintains data

4.1.20.15 Verify that interleaved block rebound to split maintains data

4.1.20.16 Verify that rebind of NULL bound block to split returns NULLs

4.1.20.17 Verify that rebind of NULL bound block to interleaved returns NULLs

4.1.20.18 Verify that rebind of NULL bound block to split maintains view integrity

4.1.20.19 Verify that rebind of NULL bound block to interleaved maintains view integrity

4.1.21 vsiptest_cblockrelease_f

4.1.21.1 Verify that release of split block returns non-NULL

4.1.21.2 Verify that release of interleaved block returns non-NULL

4.1.21.3 Attempt to release valid split released block

4.1.21.4 Attempt to release valid interleaved released block

4.1.21.5 Verify update on real portion of released split block

4.1.21.6 Verify update on imaginary portion of released split block

4.1.21.7 Verify update on interleaved released block

4.1.21.8 Attempt to release a VSIP block

4.1.22 vsiptest_cstorage

4.1.22.1 Verify that function returns value which matches VSIP_CMPLX_MEM macro

4.1.22.2 Verify that function returns a valid value

4.1.23 vsiptest_init

No output – this test nests and repeats `init()/finalize()` pairs many times.

4.2 View – VSIPL View Support Functions

4.2.1 vsiptest_CMPLX_f

4.2.1.1 Verify real part of complex scalar

4.2.1.2 Verify imaginary part of complex scalar

4.2.2 vsiptest_cmplx_f

4.2.2.1 Verify real part of complex scalar

4.2.2.2 Verify imaginary part of complex scalar

4.2.3 vsiptest_cvalldestroy_f

4.2.3.1 Attempt to create VSIP block

4.2.3.2 Attempt to bind view to VSIP block

4.2.3.3 Attempt to create complex view

4.2.3.4 Attempt to allocate arrays for user blocks

4.2.3.5 Attempt to bind split complex block

4.2.3.6 Attempt to bind view to split complex block

4.2.3.7 Attempt to bind interleaved complex block

4.2.3.8 Attempt to bind view to interleaved complex block

4.2.4 vsiptest_cvbind_f

4.2.4.1 Verify return value on failed bind attempt

4.2.4.2 Attempt to bind a view to split block

4.2.4.3 Verify data in view bound to split block

4.2.4.4 Attempt to bind a view to interleaved block

4.2.4.5 Verify data in view bound to interleaved block

4.2.4.6 Verify attributes of view bound to split block

4.2.4.7 Verify attributes of view bound to interleaved block

4.2.4.8 Attempt to bind a view to interleaved block with different attributes

4.2.4.9 Verify data in view bound to interleaved block

4.2.4.10 Verify attributes of view bound to interleaved block

4.2.4.11 Attempt to bind a view to a split block with different attributes

4.2.4.12 Verify data in view bound to split block

4.2.4.13 Verify attributes of view bound to split block

4.2.4.14 Attempt to bind a view to interleaved block with different attributes

4.2.4.15 Verify data in view bound to interleaved block

4.2.4.16 Verify attributes of view bound to interleaved block

4.2.4.17 Attempt to bind a view to a split block with different attributes

4.2.4.18 Verify data in view bound to split block

4.2.4.19 Verify attributes of view bound to split block

4.2.5 vsiptest_cvcloneview_f

4.2.5.1 Verify return value on failed clone attempt

4.2.5.2 Attempt to create an interleaved block and associated view

4.2.5.3 Attempt to create a split block and associated view

4.2.5.4 Verify that cloning an interleaved block view returns a new view

4.2.5.5 Verify that cloning a split block view returns a new view

4.2.5.6 Verify data in cloned view of interleaved block

4.2.5.7 Verify data in cloned view of split block

4.2.6 vsiptest_cvcopy_f_f

For several block sizes:

4.2.6.1 Attempt to allocated arrays

4.2.6.2 Attempt to bind split block

4.2.6.3 Attempt to bind interleaved block

4.2.6.4 Attempt to bind view to split block

4.2.6.5 Attempt to bind view to interleaved block

4.2.6.6 Attempt to create a subview of interleaved block view

4.2.6.7 Attempt to create a subview of split block view

4.2.6.8 Attempt to bind a split block

4.2.6.9 Attempt to bind a second split block

4.2.6.10 Attempt to bind an interleaved block

4.2.6.11 Attempt to bind a second interleaved block

4.2.6.12 Verify data for copying a split block view to a split block view

4.2.6.13 Verify data for copying a split block view to an interleaved block view

4.2.6.14 Verify data for copying an interleaved block view to a split block view

4.2.6.15 Verify data for copying an interleaved block view to an interleaved

block view

4.2.7 vsiptest_cvcreate_f

4.2.7.1 Verify return value on failed create

For several block sizes

4.2.7.2 Attempt to create a block and view

4.2.7.3 Check the integrity of the created block

4.2.7.4 Verify attributes on created view

4.2.7.5 Verify data in created view

4.2.8 vsiptest_cvdestroy_f

4.2.8.1 Attempt to bind a block to split data

4.2.8.2 Attempt to admit block

4.2.8.3 Attempt to bind a view to block

4.2.8.4 Attempt to destroy view

4.2.8.5 Verify return value from destroy matches block

4.2.8.6 Attempt to bind a block to interleaved data

4.2.8.7 Attempt to admit block

4.2.8.8 Attempt to bind a view to block

4.2.8.9 Attempt to destroy view

4.2.8.10 Verify return value from destroy matches block

4.2.8.11 Attempt to bind a view to a VSIP complex block

4.2.8.12 Attempt to bind a view to block

4.2.8.13 Attempt to destroy view

4.2.8.14 Verify return value from destroy matches block

4.2.9 vsiptest_cvget_f

4.2.9.1 Attempt to allocate memory

4.2.9.2 Attempt to bind interleaved block to memory

4.2.9.3 Attempt to bind split block to memory

4.2.9.4 Attempt to create VSIP complex block

4.2.9.5 Attempt to create view to whole VSIP block

4.2.9.6 Attempt to bind view to portion of VSIP block

4.2.9.7 Attempt to bind view to portion of interleaved block

4.2.9.8 Attempt to bind view to portion of split block

4.2.9.9 Verify data in view bound to VSIP block sequentially

4.2.9.10 Verify data in view bound to interleaved block sequentially

4.2.9.11 Verify data in view bound to split block sequentially

4.2.12.6 Attempt to bind view to interleaved block

4.2.12.7 Attempt to bind view to VSIP block

4.2.12.8 Verify data put to view bound to split block sequentially

4.2.12.9 Verify data put to view bound to interleaved block sequentially

4.2.12.10 Verify data put to view bound to VSIP block sequentially

4.2.12.11 Verify data put to view bound to split block randomly

4.2.12.12 Verify data put to view bound to interleaved block randomly

4.2.12.13 Verify data put to view bound to VSIP block randomly

4.2.13 vsiptest_cvputattrib_f

4.2.13.1 Attempt to allocate memory

4.2.13.2 Attempt to bind block to interleaved data

4.2.13.3 Attempt to bind block to split data

4.2.13.4 Attempt to create VSIPL complex block

4.2.13.5 Attempt to create view to all of VSIPL block

4.2.13.6 Attempt to create view to portion of VSIPL block

4.2.13.7 Attempt to create view to portion of interleaved block

4.2.13.8 Attempt to create view to portion of split block

4.2.13.9 Attempt to set and verify attributes of view bound to split block

4.2.13.10 Attempt to set and verify attributes of view bound to interleaved block

4.2.13.11 Attempt to set and verify attributes of view bound to VSIPL block

4.2.13.12 Verify data in modified view bound to VSIPL block

4.2.13.13 Verify data in modified view bound to interleaved block

4.2.13.14 Verify data in modified view bound to split block

4.2.14 vsiptest_cvputlength_f

4.2.14.1 Attempt to allocate memory

4.2.14.2 Attempt to bind block to interleaved data

4.2.14.3 Attempt to bind block to split data

4.2.14.4 Attempt to create VSIPL complex block

4.2.14.5 Attempt to create view to all of VSIPL block

4.2.14.6 Attempt to bind view to VSIPL block

4.2.14.7 Attempt to bind view to interleaved block

4.2.14.8 Attempt to bind view to split block

4.2.14.9 Attempt to set and verify length of view bound to VSIPL block

4.2.14.10 Attempt to set and verify length of view bound to interleaved block

4.2.14.11 Attempt to set and verify length of view bound to split block

4.2.14.12 Verify data in modified view to VSIPL block

4.2.14.13 Verify data in modified view to interleaved data

4.2.14.14 Verify data in modified view to split data

4.2.15 vsiptest_cvputoffset_f

4.2.15.1 Attempt to allocate memory

4.2.15.2 Attempt to bind block to interleaved data

4.2.15.3 Attempt to bind block to split data

4.2.15.4 Attempt to create VSIPL complex block

4.2.15.5 Attempt to create view to all of VSIPL block

4.2.15.6 Attempt to bind view to VSIPL block

4.2.15.7 Attempt to bind view to interleaved block

4.2.15.8 Attempt to bind view to split block

4.2.15.9 Attempt to set and verify offset of view bound to VSIPL block

4.2.15.10 Attempt to set and verify offset of view bound to interleaved block

4.2.15.11 Attempt to set and verify offset of view bound to split block

4.2.15.12 Verify data in modified view to VSIPL block

4.2.15.13 Verify data in modified view to interleaved data

4.2.15.14 Verify data in modified view to split data

4.2.16 vsiptest_cvputstride_f

4.2.16.1 Attempt to allocate memory

4.2.16.2 Attempt to bind block to interleaved data

4.2.16.3 Attempt to bind block to split data

4.2.16.4 Attempt to create VSIPL complex block

4.2.16.5 Attempt to create view to all of VSIPL block

4.2.16.6 Attempt to bind view to VSIPL block

4.2.16.7 Attempt to bind view to interleaved block

4.2.16.8 Attempt to bind view to split block

4.2.16.9 Attempt to set and verify stride of view bound to VSIPL block

4.2.16.10 Attempt to set and verify stride of view bound to interleaved block

4.2.16.11 Attempt to set and verify stride of view bound to split block

4.2.16.12 Verify data in modified view to VSIPL block

4.2.16.13 Verify data in modified view to interleaved data

4.2.16.14 Verify data in modified view to split data

4.2.17 vsiptest_cvsubview_f

4.2.17.1 Verify return value on failure

4.2.17.2 Verify data in subview

4.2.17.3 Verify data in super-view after modifying subview

4.2.17.4 Verify attributes of subview

4.2.17.5 Attempt to create subview of subview

4.2.17.6 Verify attributes of subview of subview

4.2.17.7 Verify data in subview of subview

4.2.17.8 Verify super-view data after modifying subview of subview

4.2.18 vsiptest_imag_f

4.2.18.1 Attempt to create complex block and view

4.2.18.2 Verify value returned from function for each element of view

4.2.19 vsiptest_real_f

4.2.19.1 Attempt to create complex block and view

4.2.19.2 Verify value returned from function for each element of view

4.2.20 vsiptest_valldestroy_f

For many iterations

4.2.20.1 Attempt to create a VSIPL block

4.2.20.2 Attempt to bind a view to VSIPL block

4.2.20.3 Attempt to create VISPL block and view

4.2.20.4 Attempt to create user block

4.2.20.5 Attempt to bind view to user block

4.2.20.6 Call function on all views

4.2.21 vsiptest_vbind_f

4.2.21.1 Verify return value of failure

4.2.21.2 Attempt to bind view to user block

4.2.21.3 Verify data in view

4.2.21.4 Verify attributes of view

4.2.21.5 Attempt to bind view to user block with different attributes

4.2.21.6 Verify data in view

4.2.21.7 Verify attributes of view

4.2.21.8 Attempt to bind view to user block with different attributes

4.2.21.9 Verify data in view

4.2.21.10 Verify attributes of view

4.2.21.11 Attempt to bind view to user block with different attributes

4.2.21.12 Verify data in view

4.2.21.13 Verify attributes of view

4.2.21.14 Attempt to bind view to user block with different attributes

4.2.21.15 Verify data in view

4.2.21.16 Verify attributes of view

4.2.21.17 Attempt to bind view to user block with different attributes

4.2.21.18 Verify data in view

4.2.21.19 Verify attributes of view

4.2.21.20 Attempt to bind view to user block with different attributes

4.2.21.21 Verify data in view

4.2.21.22 Verify attributes of view

4.2.22 vsiptest_vbind_l

4.2.22.1 Verify return value of failure

4.2.22.2 Attempt to bind view to user block

4.2.22.3 Verify data in view

4.2.22.4 Verify attributes of view

4.2.22.5 Attempt to bind view to user block with different attributes

4.2.22.6 Verify data in view

4.2.22.7 Verify attributes of view

4.2.22.8 Attempt to bind view to user block with different attributes

4.2.22.9 Verify data in view

4.2.22.10 Verify attributes of view

4.2.22.11 Attempt to bind view to user block with different attributes

4.2.22.12 Verify data in view

4.2.22.13 Verify attributes of view

4.2.22.14 Attempt to bind view to user block with different attributes

4.2.22.15 Verify data in view

4.2.22.16 Verify attributes of view

4.2.22.17 Attempt to bind view to user block with different attributes

4.2.22.18 Verify data in view

4.2.22.19 Verify attributes of view

4.2.22.20 Attempt to bind view to user block with different attributes

4.2.22.21 Verify data in view

4.2.22.22 Verify attributes of view

4.2.23 vsiptest_vcloneview_f

4.2.23.1 Verify return value for function failure

4.2.23.2 Attempt to create a block and view

4.2.23.3 Verify function returns a new view

4.2.23.4 Verify data in new view

4.2.24 vsiptest_vcopy_f_f

4.2.24.1 Attempt to create a vector view

4.2.24.2 Attempt to create target subview

4.2.24.3 Attempt to create source subview

4.2.24.4 Verify data in target subview after calling function

4.2.25 vsiptest_vcopy_f_l

For several block sizes:

4.2.25.1 Attempt to allocate memory for block

4.2.25.2 Attempt to create a vector view

4.2.25.3 Attempt to create target subview

4.2.25.4 Attept to create source view

4.2.25.5 Verify data in target view after calling

4.2.26 vsiptest_vcopy_i_f

For several block sizes:

4.2.26.1 Attempt to allocate memory for block

4.2.26.2 Attempt to create a vector view

4.2.26.3 Attempt to create target subview

4.2.26.4 Attept to create source view

4.2.26.5 Verify data in target view after calling

4.2.27 vsiptest_vcreate_f

For several block sizes:

4.2.27.1 Verify return value on failure

4.2.27.2 Attempt to create view

4.2.27.3 Check the integrity of the created view

4.2.27.4 Verify attributes of created view

4.2.27.5 Verify data in created view

4.2.28 vsiptest_vdestroy_f

For several block sizes:

4.2.28.1 Attempt to bind a block

4.2.28.2 Attempt to admit block

4.2.28.3 Attempt to bind a view to block

4.2.28.4 Attempt to destroy block

4.2.29 vsiptest_vdestroy_l

4.2.29.1 Attempt to bind a block

4.2.29.2 Attempt to admit block

4.2.29.3 Attempt to bind a view to block

4.2.29.4 Attempt to destroy block

4.2.30 vsiptest_vget_f

4.2.30.1 Attempt to allocate memory

4.2.30.2 Attempt to bind block

4.2.30.3 Attempt to create VSIPL block

4.2.30.4 Attempt to create view to all of VSIPL block

For several view configurations:

4.2.30.5 Attempt to create view to VSIPL block

4.2.30.6 Attempt to create view to user block

4.2.30.7 Verify data in view to VSIPL block sequentially

4.2.30.8 Verify data in view to user block sequentially

4.2.30.9 Verify data in view to VSIPL block randomly

4.2.30.10 Verify data in view to user block randomly

4.2.31 vsiptest_vgetattrib_f

4.2.31.1 Attempt to allocate memory

4.2.31.2 Attempt to bind block

4.2.31.3 Attempt to create VSIPL block

For several view configurations:

4.2.31.4 Attempt to create view to VSIPL block

4.2.31.5 Attempt to create view to user block

4.2.31.6 Verify attributes for view bound to VSIPL block

4.2.31.7 Verify attributes for view bound to user block

4.2.32 vsiptest_vgetattrib_l

4.2.32.1 Attempt to allocate memory

4.2.32.2 Attempt to bind block

4.2.32.3 Attempt to create VSIPL block

For several view configurations

4.2.32.4 Attempt to create view to VSIPL block

4.2.32.5 Attempt to create view to user block

4.2.32.6 Verify attributes for view bound to VSIPL block

4.2.32.7 Verify attributes for view bound to user block

4.2.33 vsiptest_vgetblock_f

4.2.33.1 Attempt to allocate memory

4.2.33.2 Attempt to bind block

4.2.33.3 Attempt to create VSIPL block

4.2.33.4 Attempt to bind view to all of VSIPL block

4.2.33.5 Attempt to bind view to all of user block

4.2.33.6 Attempt to allocate more memory

4.2.33.7 Attempt to bind complex block to split data

4.2.33.8 Attempt to bind complex block to interleaved data

4.2.33.9 Attempt to create VSIPL complex block

4.2.33.10 Attempt to bind views to each of the complex blocks

4.2.33.11 Verify return value for user block

4.2.33.12 Verify return value for VSIPL block

4.2.33.13 Verify return value for real part of interleaved complex block

4.2.33.14 Verify return value for real part of split complex block

4.2.33.15 Verify return value for real part of VSIPL complex block

4.2.33.16 Verify return value for imaginary part of interleaved complex block

4.2.33.17 Verify return value for imaginary part of split complex block

4.2.33.18 Verify return value for imaginary part of VSIPL complex block

4.2.34 vsiptest_vimagview_f

4.2.34.1 Attempt to allocate memory

4.2.34.2 Attempt to create split and interleaved complex blocks

4.2.34.3 Verify return value on failure

4.2.34.4 Attempt to bind view to interleaved block

4.2.34.5 Attempt to bind view to split block

4.2.34.6 Verify data in view derived from split block

4.2.34.7 Verify data in view derived from interleaved block

4.2.34.8 Verify that block created from view derived from split block behaves properly

4.2.34.9 Verify that block created from view derived from interleaved block behaves properly

4.2.34.10 Verify that block created from view derived from split block has correct attributes

4.2.34.11 Verify that block created from view derived from interleaved block has correct attributes

4.2.35 vsiptest_vput_f

4.2.35.1 Attempt to allocate memory

4.2.35.2 Attempt to create VSIPL block

4.2.35.3 Attempt to bind a block to interleaved data

4.2.35.4 Attempt to bind a block to split data

4.2.35.5 Attempt to bind a view to interleaved block

4.2.35.6 Attempt to bind a view to split block

4.2.35.7 Attempt to bind a view to VSIPL block

4.2.35.8 Verify real data put to view bound to split block sequentially

4.2.35.9 Verify imaginary data put to view bound to split block sequentially

4.2.35.10 Verify real data put to view bound to interleaved block sequentially

4.2.35.11 Verify imaginary data put to view bound to interleaved block sequentially

4.2.35.12 Verify real data put to view bound to VSIPL block sequentially

4.2.35.13 Verify imaginary data put to view bound to VSIPL block sequentially

4.2.35.14 Verify real data put to view bound to split block randomly

4.2.35.15 Verify imaginary data put to view bound to split block randomly

4.2.35.16 Verify real data put to view bound to interleaved block randomly

4.2.35.17 Verify imaginary data put to view bound to interleaved block randomly

4.2.35.18 Verify real data put to view bound to VSIPL block randomly

4.2.35.19 Verify imaginary data put to view bound to VSIPL block randomly

4.2.36 vsiptest_vputattrib_f

4.2.36.1 Attempt to allocate memory

4.2.36.2 Attempt to bind block

4.2.36.3 Attempt to create VSIPL block

4.2.36.4 Attempt to bind view to all of VSIPL

4.2.36.5 Attempt to create view to part of VSIPL block

4.2.36.6 Attempt to create view to part of user block

4.2.36.7 Verify attributes of view bound to user block

4.2.36.8 Verify attributes of view bound to VSIPL block

4.2.36.9 Verify data in modified view to VSIPL block

4.2.36.10 Verify data in modified view to user block

4.2.37 vsiptest_vputattrib_l

4.2.37.1 Attempt to allocate memory

4.2.37.2 Attempt to bind block

4.2.37.3 Attempt to create VSIPL block

4.2.37.4 Attempt to bind view to all of VSIPL

4.2.37.5 Attempt to create view to part of VSIPL block

4.2.37.6 Attempt to create view to part of user block

4.2.37.7 Verify attributes of view bound to user block

4.2.37.8 Verify attributes of view bound to VSIPL block

4.2.37.9 Verify data in modified view to VSIPL block

4.2.37.10 Verify data in modified view to user block

4.2.38 vsiptest_vputlength_f

4.2.38.1 Attempt to allocate memory

4.2.38.2 Attempt to bind block

4.2.38.3 Attempt to create VSIPL block

4.2.38.4 Attempt to create view to VSIPL block

4.2.38.5 Attempt to create view to portion of VSIPL block

4.2.38.6 Attempt to create view to user block

4.2.38.7 Verify attributes of modified view to VSIPL block

4.2.38.8 Verify attributes of modified view to user block

4.2.38.9 Verify data in modified view to VSIPL block

4.2.38.10 Verify data in modified view to user block

4.2.39 vsiptest_vputoffset_f

4.2.39.1 Attempt to allocate memory

4.2.39.2 Attempt to bind block

4.2.39.3 Attempt to create VSIPL block

4.2.39.4 Attempt to create view to VSIPL block

4.2.39.5 Attempt to create view to portion of VSIPL block

4.2.39.6 Attempt to create view to user block

4.2.39.7 Verify attributes of modified view to VSIPL block

4.2.39.8 Verify attributes of modified view to user block

4.2.39.9 Verify data in modified view to VSIPL block

4.2.39.10 Verify data in modified view to user block

4.2.40 vsiptest_vputstride_f

4.2.40.1 Attempt to allocate memory

4.2.40.2 Attempt to bind block

4.2.40.3 Attempt to create VSIPL block

4.2.40.4 Attempt to create view to VSIPL block

4.2.40.5 Attempt to create view to portion of VSIPL block

4.2.40.6 Attempt to create view to user block

4.2.40.7 Verify attributes of modified view to VSIPL block

4.2.40.8 Verify attributes of modified view to user block

4.2.40.9 Verify data in modified view to VSIPL block

4.2.40.10 Verify data in modified view to user block

4.2.41 vsiptest_vrealview_f

4.2.41.1 Attempt to allocate memory

4.2.41.2 Attempt to create split and interleaved complex blocks

4.2.41.3 Verify return value on failure

4.2.41.4 Attempt to bind view to interleaved block

4.2.41.5 Attempt to bind view to split block

4.2.41.6 Verify data in view derived from split block

4.2.41.7 Verify data in view derived from interleaved block

4.2.41.8 Verify that block created from view derived from split block behaves properly

4.2.41.9 Verify that block created from view derived from interleaved block behaves properly

4.2.41.10 Verify that block created from view derived from split block has correct attributes

4.2.41.11 Verify that block created from view derived from interleaved block has correct attributes

4.2.42 vsiptest_vsubview_f

4.2.42.1 Verify return value on failure

4.2.42.2 Verify data in subview

4.2.42.3 Verify data in super-view after modifying subview

4.2.42.4 Verify attributes of subview

4.2.42.5 Attempt to create subview of subview

4.2.42.6 Verify attributes of subview of subview

4.2.42.7 Verify data in subview of subview

4.2.42.8 Verify data in superview after modifying subview of subview

4.3 Elwise – VSIPL vector element-wise functions

The "elwise" module contains the tests for the vector element-wise functions. For each function, the Test Suite performs the function on a preloaded, predefined set of data, and compares the results returned with a set of predefined, expected data. Any deviation in the returned results which is greater than some predefined value associated with the particular test generates an error, and causes the test to fail. The intent of this testing is not to establish an arbitrary accuracy threshold, but to test that the VSIPL implementation performs the required function. Therefore, it is permissible for implementors to distribute the Test Suite with their implementations with modified values for the accuracy thresholds. These values are reported by the Test Suite with the appropriate command line arguments.

The predefined data vectors for each function are generated by Matlab and are specifically generated for each function to generate a data set which will cover the domain of the function without imposing unnecessary arbtitrary accuracy requirements on the implementation. The source files for the Matlab programs used to generate the data are in the `matlab/` directory of the distribution package.

This section of the Test Suite also attempts to verify that the VSIPL implementation handles the various forms of memory organization for views properly. For views of real blocks these forms are:

- VSIP created real block

- Derived from real part of VSIP complex block
- Derived from imaginary part of VSIP complex block
- User real block
- Derived from real part of user split complex block
- Derived from imag part of user split complex block
- Derived from real part of user interleaved complex block
- Derived from imag part of user interleaved complex block

For complex blocks, these forms are:

- VSIP created block
- User created split block
- User created interleaved block

The Test Suite calls each function with a view to every form of block applicable for that function as an argument, and repeats the data analysis.

For each function, the Test Suite also tests in-place operation of the function, and tests the function with views to blocks with various stride combinations.

The tests in this module are as follows:

4.4 Sigproc – VSIPL Signal Processing and Random Functions

The signal processing functions in the Core Lite Profile consist of Fast Fourier Transform (FFT), Finute Impuls Response (FIR), and histogram functions. This module contains those functions as well as the random functions. The support functions for the signal processing functionality have absolutely no observable effects, and are impossible to test directly. For example, an FFT object can not be tested without calling the FFT functions. Therefore, the support functions are not tested directly, and are only tested implicitly in conjuction with the applicable signal processing functions.

The signal processing functions are tested in a similar fashion to the vector element wise functions. For each function, a set of input and expected output vectors is predefined by Matlab and is compared against the output generated by the function. Any deviation greater than the specified tolerance generates an error message and causes the test to fail. The source files for the Matlab programs used to generate the data are in the `matlab/` directory of the distribution package.

Unlike the vector element wise tests, the "sigproc" module tests do not attempt to test memory organization management, and only test the function against basic vector types. However, since the signal processing functions are more complex in nature, the Test Suite repeats the tests with several different sets of input and output for each function.

The signal processing tests are the following:

4.4.1 vsiptest_ccfftop_f

4.4.2 vsiptest_cfirflt_f

4.4.3 vsiptest_crfftop_f

4.4.4 vsiptest_firflt_f

4.4.5 vsiptest_rcfftop_f

4.4.6 vsiptest_vhisto_f

Also included in the "sigproc" module of the Test Suite are the tests of the random function

4.4.7 vsiptest_vrandu_f

For the portable random number generator case, the Test Suite verifies that the implementation produces the expected results in the same way that it verifies the functionality of the signal processing functions. A set of input parameters, and the

corresponding expected out vectors are supplied with the Test Suite, and the the Test Suite compares the values returned by the VSIPL implementation with the expected values.

For the non-portable random number generator case, the only requirement on the implementation is that it produce output values that are between 0 and 1.0 inclusive. The Test Suite generates a random vector in the non-portable case, and verifies that each value in the output vector meets this requirement.

5. Appendix A: Copyright Statement

This Program is licensed, not sold to you by GEORGIA TECH RESEARCH CORPORATION ("GTRC"), owner of all code and accompanying documentation (“Program”) for use only under the terms of this License, and GTRC reserves any rights not expressly granted to you.

1. This License allows you to:

(a) make copies and distribute copies of the Program’s source code provide that any such copy clearly displays any and all appropriate copyright notices and disclaimer of warranty as set forth in Article 5 and 6 of this License. All notices that refer to this License and to the absence of any warranty must be kept intact at all times. A copy of this License must accompany any and all copies of the Program distributed to third parties.

A fee may be charged to cover the cost associated with the physical act of transferring a copy to a third party. At no time shall the program be sold for commercial gain either alone or incorporated with other program(s) without entering into a separate agreement with GTRC.

(b) modify original copy or copies of the Program or any portion thereof (“Modification(s)”). Modifications may be copied and distributed under the terms and conditions as set forth above, provided the following conditions are met:

i) any and all modified files must be affixed with prominent notices that you have changed the files and the date that the changes occurred.

ii) any work that you distribute, publish, or make available, that in whole or in part contains portions of the Program or derivative work thereof, must be licensed at no charge to all third parties under the terms of this License.

iii) if the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to display and/or print an announcement with all appropriate copyright notices and disclaimer of warranty as set forth in Article 5 and 6 of this License to be clearly displayed. In addition, you must provide reasonable access to this License to the user.

Any portion of a Modification that can be reasonably considered independent of the Program and separate work in and of itself is not subject to the terms and conditions set forth in this License as long as it is not distributed with the Program or any portion thereof.

2. This License further allows you to copy and distribute the Program or a work based on it, as set forth in Article 1 Section b in object code or executable form under the terms of Article 1 above provided that you also either:

i) accompany it with complete corresponding machine-readable source code, which must be distributed under the terms of Article 1, on a medium customarily used for software interchange; or,

ii) accompany it with a written offer, valid for no less than three (3) years from the time of distribution, to give any third party, for no consideration greater than the cost of physical transfer, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Article 1 on a medium customarily used for software interchange; or,

3. Export Law Assurance.

You agree that the Software will not be shipped, transferred or exported, directly into any country prohibited by the United States Export Administration Act and the regulations thereunder nor will be used for any purpose prohibited by the Act.

4. Termination.

If at anytime you are unable to comply with any portion of this License you must immediately cease use of the Program and all distribution activities involving the Program or any portion thereof.

5. Disclaimer of Warranties and Limitation on Liability.

YOU ACCEPT THE PROGRAM ON AN "AS IS" BASIS. GTRC MAKES NO WARRANTY THAT ALL ERRORS CAN BE OR HAVE BEEN ELIMINATED FROM PROGRAM. GTRC SHALL NOT BE RESPONSIBLE FOR LOSSES OF ANY KIND RESULTING FROM THE USE OF PROGRAM AND ITS ACCOMPANYING DOCUMENT(S), AND CAN IN NO WAY PROVIDE COMPENSATION FOR ANY LOSSES SUSTAINED, INCLUDING BUT NOT LIMITED TO ANY OBLIGATION, LIABILITY, RIGHT, CLAIM OR REMEDY FOR TORT, OR FOR ANY ACTUAL OR ALLEGED INFRINGEMENT OF PATENTS, COPYRIGHTS, TRADE SECRETS, OR SIMILAR RIGHTS OF THIRD PARTIES, NOR ANY BUSINESS EXPENSE, MACHINE DOWNTIME OR DAMAGES CAUSED TO YOU BY ANY DEFICIENCY, DEFECT OR ERROR IN PROGRAM OR MALFUNCTION THEREOF, NOR ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED. GTRC DISCLAIMS ALL WARRANTIES, BOTH EXPRESS AND IMPLIED RESPECTING THE USE AND OPERATION OF PROGRAM AND ITS ACCOMPANYING DOCUMENTATION, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR PARTICULAR PURPOSE AND ANY IMPLIED WARRANTY ARISING FROM COURSE OF PERFORMANCE, COURSE OF DEALING OR USAGE OF TRADE. GTRC MAKES NO WARRANTY THAT PROGRAM IS ADEQUATELY OR COMPLETELY DESCRIBED IN, OR BEHAVES IN ACCORDANCE WITH ANY ACCOMPANYING DOCUMENTATION. THE USER OF PROGRAM IS EXPECTED TO MAKE THE FINAL EVALUATION OF PROGRAM'S USEFULNESS IN USER'S OWN ENVIRONMENT.

GTRC represents that, to the best of its knowledge, the software furnished hereunder does not infringe any copyright or patent.

GTRC shall have no obligation for support or maintenance of Program.

6. Copyright Notice.

THE SOFTWARE AND ACCOMPANYING DOCUMENTATION ARE COPYRIGHTED WITH ALL RIGHTS RESERVED BY GTRC. UNDER UNITED STATES COPYRIGHT LAWS, THE SOFTWARE AND ITS ACCOMPANYING DOCUMENTATION MAY NOT BE COPIED ACCEPT AS GRANTED HEREIN.

You acknowledge that GTRC is the sole owner of Program, including all copyrights subsisting therein. Any and all copies or partial copies of Program made by you shall bear the copyright notice set forth below and affixed to the original version or such other notice as GTRC shall

designate. Such notice shall also be affixed to all improvements or enhancements of Program made by you or portions thereof in such a manner and location as to give reasonable notice of GTRC's copyright as set forth in Article 1.

Said copyright notice shall read as follows: