

DFTVIEW

**STIL and WGL Graphical
Display and Validation Tool**

User Manual

**Release 2.0.3
2012**





1	OVERVIEW.....	5
2	SETUP	7
2.1	DFTView Install Script.....	7
2.1.1	Install Mode.....	7
2.1.2	Check Mode.....	7
2.2	Uninstalling DFTView.....	8
2.3	Manual Installation and Setup	8
2.4	Waveform Viewer Selection.....	9
2.5	Licensing.....	10
2.5.1	Node Locked License	10
2.5.2	Floating License	11
2.5.3	Setting up a license server:.....	12
2.5.4	Limited Demo Mode.....	12
2.6	Additional Requirements	12
3	DFTView INVOCATION	14
3.1	Running DFTView	14
3.2	DFTView Demo Mode	15
3.3	Running in Demo Mode	16
3.4	Command File.....	17

4	DFTView OPERATION: Normal Mode.....	19
4.1	Overview	19
4.1.1	Source Editor	19
4.2	Button Controls	20
4.2.1	Regenerate Display.....	20
4.2.2	Adjust Waveform Display	20
4.2.3	Previous Vector	20
4.2.4	Next Vector.....	20
4.2.5	Adjust Source Display	21
4.2.6	Terminate Session	21
4.3	Menu Options.....	21
4.3.1	File Menu.....	21
4.3.2	Edit Menu	23
4.3.3	Help Menu	24
4.4	Error Detection.....	24
4.5	Separating Bidirectional Pins.....	25
5	DFTView OPERATION: V-mode	26
5.1	Overview	26
5.2	Vertical Search	28
5.3	Vertical Search/Replace	28
5.4	Locate Pin/Signal.....	29
5.5	Smash and Compress.....	29
5.6	Other Operations	30
5.7	Pin Selection and Formatting.....	30

5.7.1	File Format.....	31
5.7.2	Named Pin Groups.....	31
5.7.3	Display Radix.....	32

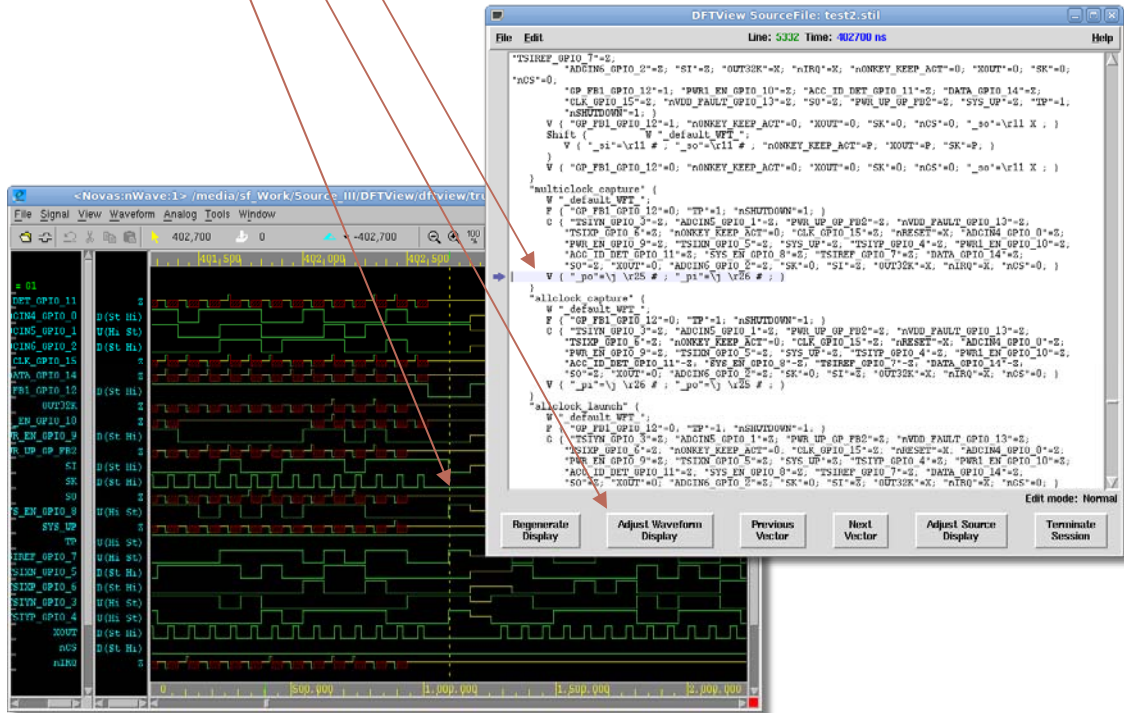
1 OVERVIEW

The DFTView program allows the user to view and edit a source file in WGL, STIL or VCD/EVCD format, or a vtran-generated Verigy 93000 or Teradyne FLEX source file in a text editor window, while viewing the waveforms represented by the source file timing and vectors in a SpringSoft **nWave** or a **GTKWave** window. It provides the following capabilities:

- ◆ Directly view the waveforms that will be applied to a device on a tester as described in a WGL, STIL, VCD/EVCD file or a vtran-generated Verigy 93000 or Teradyne FLEX test vector file.
- ◆ Waveform viewing of these files provides exact state as well as edge timing information for the test program in a very familiar and graphical format.
- ◆ Automatically check a source file for syntax errors, and validate by inspection a source file's contents prior to the test vectors being applied to a device on the tester.
- ◆ Edit the contents of the source file and see the exact effect on tester waveforms. This is a common situation for debug and SOC integration.
- ◆ Some testers directly accept STIL files as test programs so this allows the user to directly view and validate these test patterns.
- ◆ Select a Vector line in the source file with your cursor, click on the Adjust Waveform Display button, and see the exact corresponding waveform location in the waveform display window.
- ◆ Select a waveform location with your cursor, click on the Adjust Source Display button, and see the corresponding source vector line in the source display window.

Source File to Waveform:

1. Place cursor on desired vector statement
2. Click "Adjust Waveform" button
3. Waveform cursor identifies location



2 SETUP

2.1 DFTView Install Script

As outlined below in DFTView can be installed manually without much trouble, most of the steps involve setting up your environment variables correctly. To facilitate this process we have included an installer script which will guide you through installation and setup. The script *install.sh* can be run in two modes: the default “install mode” (Section 2.1.1) which will help you install **DFTView**, and a “check mode” (Section 2.1.2) which will examine your current system and environment setup to make sure everything is in order for you to use DFTView.

2.1.1 Install Mode

In this default mode simply invoke *install.sh* without arguments, and it will present you with a short set of questions to help guide you through the installation process. Most questions will have a default answer you can accept by just hitting <Enter>. As the script runs it will prompt you for some information regarding your installation directory, preferred waveform viewer and so on. It will then install the required files for DFTView, and then present you with a few lines to add to your `.profile`, `.cshrc` or other startup script, already customized for your installation. Even if you decide to leave the DFTView package where you unpacked it, you should still run *install.sh* and accept the default value for the installation directory, as this will still help you to configure your environment correctly.

During the installation process the script will check to see if you have a compatible waveform viewer already installed on your system - see "**Waveform Viewer Selection**" (Section 2.4). If no compatible viewer is detected, *install.sh* will offer you the option to download a GTKWave binary package directly from the Source III site and install it for you. This is an unmodified build of GTKWave version 3.3.21 which includes the complete source code and Tcl/Tk runtime environment. If for any reason the package cannot be downloaded automatically, you can download it yourself from the Source III FTP site and place it in the DFTView “support” directory (you don't have to uncompress or unpack it). The next time you run the installer it will automatically install GTKWave from this downloaded file.

When the installer has finished, it will display the environment variables you need to add to your shell login scripts (a set of each of C shell and Bourne shell lines will be displayed).

2.1.2 Check Mode

The *install.sh* script also provides a “check mode” where it will examine your system and environment to determine if all the prerequisites for DFTView are available. Simply run as follows:

```
./install.sh -check
```

It will report back on required system components and important environment variables, with suggestions on what to do if anything appears amiss.

Running the installer in this check mode can be performed any time, and does not require that you installed DFTView via *install.sh* to begin with. This may be useful for example if you are a systems administrator and need to verify a custom environment setup script for your users.

2.2 Uninstalling DFTView

We hate to lose you, but if you wish to completely uninstall DFTView from your system there is no special command or procedure required. Simply delete the contents of the `$DFTVIEW_ROOT` directory, and optionally remove any environment configurations you set up as part of the installation.

2.3 Manual Installation and Setup

The DFTView program bundles are available for Solaris SPARC and Linux platforms. These can be downloaded from the Source III web site at: <http://www.sourceiii.com/support.php>, select the DOWNLOAD SOFTWARE tab. Once the bundle is downloaded, gunzip it and untar it into a directory. Then do the following:

- 1) Define an environment variable named `DFTVIEW_ROOT` which points to this directory. This is used by the application programs to find additional program modules. In order to set the `DFTVIEW_ROOT` environment variable, add the following command to your `.cshrc` and/or `.login` file:

```
setenv DFTVIEW_ROOT /path
```

where `/path` is the directory path into which this software was loaded. For example, if the tar file for linux64 was loaded in the directory `/usr/source3`, then the following command would be used:

```
setenv DFTVIEW_ROOT /usr/source3/linux64
```

You then must either logout and log back in, or issue the following command from the console:

```
source ~/.cshrc
```

-or-

```
source ~/.login
```

NOTE: The instructions above are geared for a csh (C shell). Setup will vary slightly for different shells.

- 2) Add the /path to your search path so the DFTView program and modules can be found. If you installed GTKWave from the Source III FTP site using the installer, you should also add \$DFTVIEW_ROOT/support/ to the **end** of your path, e.g.:

```
setenv PATH ${PATH}:%DFTVIEW_ROOT/support
```

2.4 Waveform Viewer Selection

DFTView can be run using either the SpringSoft nWave (Verdi) waveform display tool, or the open-source GTKWave waveform display tool. The waveform tool to be used is selected using an environment variable named DFTVIEW_DISPLAY. To select the GTKWave tool, use:

```
setenv DFTVIEW_DISPLAY GTKWave
```

For optimal performance with GTKWave, it is recommended that you use a recent release such as v3.3 or later. If GTKWave is already installed on your system you don't need to do anything else. If it doesn't appear to be installed (e.g. "which gtkwave" doesn't return a path to the gtkwave executable), you can install the package on Linux with the "yum" or "apt-get" commands as applicable, e.g.:

RedHat, CentOS, Fedora, etc.: `sudo yum install gtkwave`

Ubuntu, Debian, etc: `sudo apt-get install gtkwave`

We have also prepared Linux and Solaris-compatible GTKWave binary packages (release 3.3.21) which are freely available from our support FTP site. Installation is as simple as running the *install.sh* script as described in "**DFTView Install Script**" (Section 2.1) or downloading the relevant package compatible with your system, unpacking it and adding the path to the "gtkwave" command to your PATH. At this stage you can verify GTKWave is working just by typing the command "gtkwave".

To select the SpringSoft nWave waveform display tool, use:

```
setenv DFTVIEW_DISPLAY nWave
```

If no DFTVIEW_DISPLAY environment variable is defined, then the SpringSoft nWave tool is assumed. When the nWave tool is being used to display waveforms, by default, DFTView

launches nWave directly. However, for some customer environments it is necessary to launch Verdi first and then bring-up nWave from Verdi. In order to address this, an environment variable named DFTV_VERDI can be defined and set to a 1, which will cause Verdi to be launched prior to nWave. To activate this, use:

```
setenv DFTV_VERDI 1
```

2.5 Licensing

The DFTView program uses Flexera (FlexLM) licensing software. We support both floating and node-locked licenses for evaluation and purchase. Depending on the licensing model you choose, we will need some basic information from you so that we can provide you with the appropriate license.

2.5.1 Node Locked License

*Summary: Run **lmhostid** on each target system where you wish to run DFTView*

In the simplest case, we can issue you a node-locked license. This, as the name suggests, “locks” the execution of DFTView to a specific system, or node. For this license all we need is the host ID which you can get by running the Flexera utility called *lmhostid* which we have provided in the support/ directory of each DFTView bundle. Simply run the command without any arguments and send us the output which will look something like this (with a different host ID, of course):

```
lmhostid - Copyright (c) 1989-2010 Flexera Software, Inc. All
Rights Reserved.
```

```
The FLEXnet host ID of this machine is "123456789abc"
```

If you wish to have several systems included in a node-locked license (thus allowing DFTView to be run on more than one workstation) simply run the *lmhostid* utility on each system and send us the output from all of them.

When you run DFTView it will examine the license file we will issue for you, and compare the host ID defined in that file with the current host ID (determined automatically at run time). DFTView locates the license file by examining an environment variable named LM_LICENSE_FILE. Just set this environment variable to where-ever you install your license file, e.g.:

```
export LM_LICENSE_FILE=/foo/bar/sourceIII.lic          (bash, zsh)
setenv LM_LICENSE_FILE /foo/bar/sourceIII.lic         (csh, tcsh)
```

Both examples assume you have only a single license file – the one we've provided you with. In cases where you already have a `LM_LICENSE_FILE` environment variable defined just add the new license file name to the end, separated from the other entries with a colon, e.g.:

```
export LM_LICENSE_FILE=/foo/file1.lic:/foo/bar/sourceIII.lic
```

2.5.2 Floating License

*Summary: Run **lmhostid** on the license server system and optionally provide the server hostname*

A floating license provides a centralized way to provide a license for more than one system. Rather than needing to record and track host IDs for potentially dozens or hundreds of individual systems, FlexLM can manage “handing out” licenses from a central pool. In this model, only the host ID of the server system itself is required, so we only need the output of *lmhostid* and optionally the server hostname. If you provide the hostname we can add that to the license file for you, otherwise simply replace the generic 'server_hostname' with your server hostname on the SERVER line in the license file. You must also change the VENDOR line in the license file we provide so that it correctly identifies the full location of the *siid* license daemon (also included in the DFTView bundle).

With a floating license you can make the license file available to each workstation (called a client) on which DFTView will run, perhaps by installing on a shared filesystem, or by copying it to the same location on each client system. Then add the location of the floating license file to the `LM_LICENSE_FILE` variable, just as described in the previous section on node-locked licenses:

```
export LM_LICENSE_FILE=/foo/bar/sourceIII.lic           (bash, zsh)
setenv LM_LICENSE_FILE /foo/bar/sourceIII.lic         (csh, tcsh)
```

If you are using the SpringSoft nWave viewer in conjunction with DFTView, it is most likely that you already have a license server system up and running at your site. In this case, a floating license is better suited for your setup, and by providing us with the same server hostname and host ID currently in use we can provide you with a suitable license you can incorporate into your current environment. FlexLM allows license files to be combined provided the number of SERVER lines in both files match, and the host ID on each SERVER line is identical. In this way a single FlexLM server system can manage multiple licenses for different products.

2.5.3 Setting up a license server:

If you wish to request a floating license, but do not currently have a license server set up, you will need to run Flexera's *lmgrd* or *lmadmin* programs on a designated server system. We have included a compatible *lmgrd* daemon in the DFTView bundle, in the support/ directory. At a minimum you would run *lmgrd* and point it to a suitable license file:

```
lmgrd -c /path/to/license/file.lic
```

Note: It is **not** advised to run *lmgrd* as root.

There are many other possible configuration options when setting up a floating license server. If you are interested in more advanced topics in FlexLM license administration refer to the detailed License Administration Guide which is available on the Flexera Software support websites:

<http://www.flexerasoftware.com/support/>

http://www.globes.com/support/fnp_utilities_download.htm#docs

2.5.4 Limited Demo Mode

DFTView can also be run in DEMO mode (Section 3.2), with no need for a license. This mode is entered automatically if you do not have a FlexLM license file installed.

2.6 Additional Requirements

To run DFTView your system must also have the following:

- ◆ The "wish" interpreter (Tcl/Tk) must be installed. Tcl/Tk is included by default with most Linux distributions. You can also download a recent version directly from <http://tcl.tk/> if necessary. For Solaris we recommend downloading a pre-compiled binary version of Tcl/Tk from <http://www.sunfreeware.com/>. Regardless of your chosen method for installing Tcl/Tk, make sure you install, or already have available, **version 8.0.5 or newer (8.4 or newer recommended)**.
- ◆ If using nWave, a SpringSoft Verdi license must be available and the "vfast" utility and "nWave" programs must be in your path.
- ◆ If using GTKWave, the "gtkwave" program must be in your path.

Once the above setup has been performed, you should be able to go to any working directory and run DFTView on WGL, STIL, VCD/EVCD and vtran-generated Verigy 93000 and Teradyne FLEX files.

3 DFTView INVOCATION

3.1 Running DFTView

There are two options to invoke DFTView. With the first option, the DFTView command line parameters include the source file name(s), and an optional units specification. The syntax is:

```
DFTView [-wgl|-stil|-93k|-flex] [infile] [timing_file] [units]
```

Note that when viewing Verigy 93K files, both a vector file (.avc or .hptab) and a timing file (.dvc) is required. Likewise for reading a FLEX file, both a vector file (.atp or .flex) and a timing file are needed. For FLEX, the timing file is a base name. The Reader invoked by DFTView will append the suffix `_esets.txt` and `_tsets.txt` to the base name to find the edge sets and time sets FLEX timing files. It is not necessary to invoke DFTView with any command line arguments. It can be invoked with simply:

```
DFTView
```

This will bring-up an empty waveform and DFTView source window. From here, you can use the **File→Open** drop-down menu to load a new file. See the discussion on using this feature below.

Using another option, the DFTview command line can specify a command file, which specifies the input parameters. The optional command file is described below. The syntax for invoking DFTView with a command file is:

```
DFTView -cmd cmdfilename
```

See the discussion below for using a DFTView command file. Compressed WGL and STIL files (“gzipped”) are supported and are uncompressed (“unzipped”) on the fly at load time.

The units parameter can be 1ps, 10ps, 100ps or 1ns. If the file name extension is “.wgl”, “.stil”, “.avc”, “.atp”, or “.flex”, the `-wgl`, `-stil`, `-93k` or `-flex` switch is not needed. The user must have write access to the directory containing the source file.

Some command line examples for invoking DFTView are:

```
DFTView
DFTView ovf.wgl
DFTView ovf.stil
DFTView -wgl myWglFile
DFTView -stil myStilFile
DFTView -wgl ovf.wgl.gz
DFTView -stil ovf.stil.gz
```

```
DFTView -93k maximo.avc maximo.dvc
DFTView -93k sample.hptab sample.dvc -vmode
```

DFTView begins by converting the source file to a waveform viewing file for either nWave or GTKWave. It then automatically loads the waveforms in the display program and all signals are selected. In addition the contents of the source vector file are loaded in the DFTView text editor. Once the source file is loaded and displayed in the DFTView window, you can begin viewing and editing.

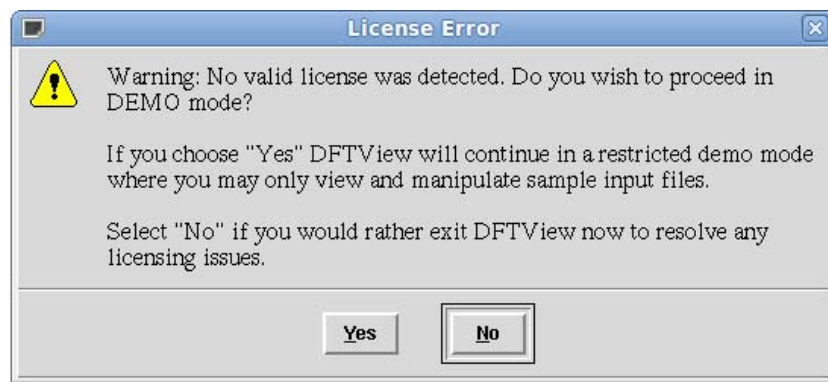
Note that vector files can be gzipped and will be expanded automatically, but vector files **must** be provided uncompressed (Section 4.3.1).

You can, if loading 93k format vector and timing files on the command line, also provide an optional `-vmode` argument which will put the editor directly into V-mode (Section 5). The `-vmode` argument is ignored for other file input formats.

The nWave or GTKWave window is used to scroll through signals and event times to view the waveforms. Refer to the SpringSoft documentation for using nWave or the documentation for GTKWave for its use.

3.2 DFTView Demo Mode

When you run DFTView, if you have not purchased a license key for DFTView, or if there is an issue with the license, a dialog box will appear giving you the option of running in DEMO mode for evaluation, as shown here:



If this dialog appears and you *have* purchased a license or have received an evaluation license, you can select “No” here to exit the application and address whatever license issue you may have – check the parent terminal window where you launched DFTView for specific license-related messages.

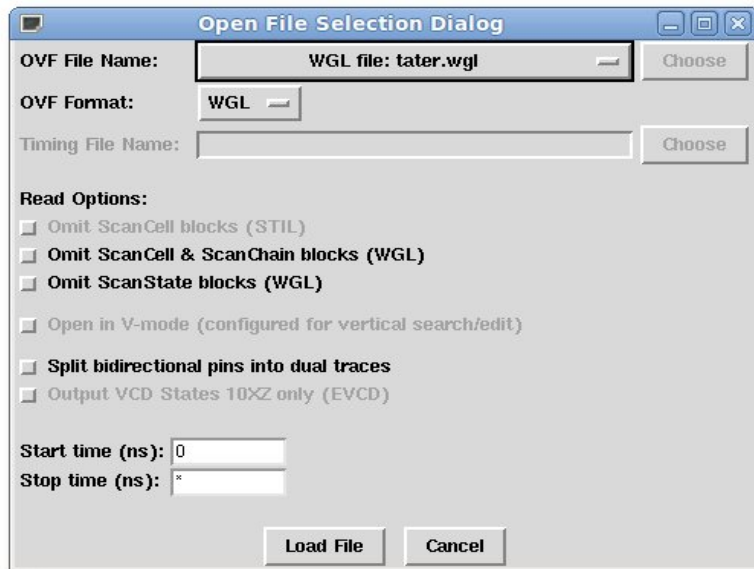
If you select “Yes”, you will enter DEMO mode. In this mode, DFTView will allow you to view and interact with a set of demo files shipped with the application, but will not allow you to view, edit or save your own files. Also note that if DFTView enters DEMO mode, any command line arguments you may have passed when launching the application (e.g. an input filename or format) will be ignored.

You can always tell if DFTView is running in DEMO mode - it will be clearly indicated at the bottom right of the main application window:



3.3 Running in Demo Mode

As with normal (licensed) mode, you can select **File**→**Open** to select an input file. In DEMO mode however, the file open dialog box will appear slightly different to the regular licensed version, as shown below – The “Choose” buttons will be disabled and the OVF File Name will not allow you to type in a file name. Instead, you will be presented with a drop-down list of demo files (which are shipped with the application and reside in the \$DFTVIEW_ROOT/demo/ directory). Other options such as Omitting ScanCell and ScanChain blocks, start and stop time etc. are available if you wish to try them out.



The OVF Format will be set for you each time you select a new demo input file, although you may select it manually if you wish. If you select a demo OVF file which also requires a Timing file, that field will be set for you automatically.

Once you've selected and opened a demo file, it will be displayed in the main application window and in the waveform viewer (if configured). All the normal controls for navigating around the file will be available to you, and you can also make limited edits to the input file to see the resulting changes in the waveforms.

If the drop-down list of OVF file names appears to be empty, verify that your `$DFTVIEW_ROOT/demo/` directory and the contents therein are readable. Also note that if you add other files to the `demo/` directory, they will **not** be readable by DFTView in DEMO mode.

In addition, you can view the provided 93k demo files in V-mode (Section 5). Once opened in V-mode feel free to make minor changes to the file using the various operations in Section 5 as a guide. Of particular note, we have provided a sample pin-formatting file `sbc.example.fmt` which can be applied to the `sbc2.avc` and `sbv2.dvc` files if opened in V-mode. Select the **Load V-mode format file** operation from the **Edit** menu and select the example format file. You can also view this file in a standard text editor such as `vim`, `nedit` or `emacs`.

3.4 Command File

In order to provide more flexibility in passing parameters and using scripts to drive the viewing, an optional command file can be passed to DFTView as described above. This section discusses the contents of a DFTView command file. All of these features can also be accessed from the DFTView GUI in interactive mode.

Long load times for large WGL/STIL files can occur due to the flattening of cycle-based files to event streams for waveform display, especially when scan data is present. DFTview has been enhanced to allow the user the option to specify a subset of cycle times. In the future, it will be enhanced to allow the user the option to specify a subset of signals. The Reader generates a waveform display file that only includes the data for the specified time range.

The `START_TIME` and `STOP_TIME` commands define the first and last event times to be included in the waveform display file. If `START_TIME` is not specified, it defaults to 0ns. If `STOP_TIME` is not specified, it defaults to the last event in the input vector file. The syntax is:

```
START_TIME n;
STOP_TIME n;
```

where `n` is a positive integer or real number.

The command line arguments also are specified in the command file, when a command file is used. The syntax is:

```
FORMAT wgl|stil|93k;  
INFILE <file name>;  
AUXFILE <filename.dvc>;  
UNITS <units>;
```

As on the command line, the FORMAT and UNITS commands are optional. An example of a DFTview command file, named runDFTview.cmd, is:

```
INFILE myDesign.stil;  
START_TIME 100;  
STOP_TIME 50000;
```

To invoke DFTview using this command file, the command line is:

```
DFTview -cmd runDFTview.cmd
```

The waveform file created by the Reader will only contain events from the STIL file starting at 100ns and ending at 50000ns, inclusive.

Tip: When loading files which will generate large numbers of vectors (millions or more) you may find loading time to be longer than expected. DFTView flattens loops and repeats when generating the vectors so that they can be displayed, and this can take some time. It's highly recommended that you avail of the START_TIME and STOP_TIME feature which allows you to quickly load an area of interest without having to load every vector. If you don't wish to use a command file you can also specify the start and stop times via the File/Open dialog (see section 4.3.1).

4 DFTView OPERATION: Normal Mode

4.1 Overview

When you launch DFTView you will see a source file editing window and, depending on which waveform application you configured, a waveform viewer window. The latter is a separate application which DFTView will launch on your behalf. The main source editing window contains a number of menu operations as well as some button controls along the bottom of the window. In summary these various operations and controls are:

- **Regenerate Display (button)**
- **Adjust Waveform Display (button)**
- **Previous Vector (button)**
- **Next Vector (button)**
- **Adjust Source Display (button)**
- **Terminate Session (button)**
- **File (drop-down menu)**
 - File → Open
 - File → Close
 - File → Regenerate Display
 - File → Save Source
 - File → Exit
- **Edit (drop-down menu)**
 - Edit → Go To Line Number
 - Edit → Go to Time Point
 - Edit → Search
- **Help (drop-down menu)**
 - Help → About
 - Help → File → *selection*
 - Help → Edit → *selection*
 - Help → Buttons → *selection*

4.1.1 Source Editor

The main portion of the DFTView display window is the text editor itself. You can make edits to your source and then click the **Regenerate Display** button to see how your edit will affect the waveform (Section 4.2.1). You can also elect to save your modified source (Section 4.3.1). Note that for some formats, especially those which require a separate timing file, edits which add new signals are not recommended, and in fact may cause DFTView to detect these additions as errors.

4.2 Button Controls

4.2.1 Regenerate Display

The **Regenerate Display** button is used to update the data displayed in the waveform window, after modifying the source file vector or timing data with the DFTView editor. This button does NOT update the original source file specified on the command line or in the command file. In some circumstances when you make changes to your source file, you may see this button begin to flash. This occurs as a reminder that the waveform and source files are now possibly out of sync, and you should click this button once you have completed any additional edits you wish to make. You will usually see this button flash after you have performed a *Smash* or *Compress* operation (section 5.3).

4.2.2 Adjust Waveform Display

The **Adjust Waveform Display** button is used to synchronize the waveform display with the DFTView source editor display. This can be used in several ways:

- ◆ When clicked on a Procedure Call, Macro, Main Pattern Loop/Repeat, or Main Pattern Scan statement in the Source Display, causes the waveform viewer and the text editor's cursors to move to the first cycle of the pattern structure; W and C statements, for example, are skipped. Repeated clicks act as Next Cycle until the end of the structure. When clicked at the end, causes a return to first cycle of the pattern structure.
- ◆ When clicked on a Loop or STIL Shift inside a Macro or Procedure, the behavior is the same as Loop in the Main Pattern, except clicking on last (end) cycle results in falling out of the Loop or Shift.

DFTView sets the waveform cursor to adjust the display. Any user setting of the cursor will be overridden. For a multiple-line vector in the source file editor, the editor's cursor must be set to the first line of the vector.

4.2.3 Previous Vector

The **Previous Vector** button provides a means to step sequentially backwards through vector statements in the source file while also moving the waveform display to the corresponding waveform location.

4.2.4 Next Vector

Steps sequentially forward through vectors. Repeated clicks before or during a procedure, Macro, or Loop/Repeat in the Source Display causes the waveform viewer and the text editor's cursors to move through the pattern structure. When clicked at end of a pattern structure,

causes the cursors to advance to vector related statements following the procedure, macro, or loop/repeat pattern structure.

When clicked on a Scan or STIL Shift statement, causes the Scan/Shift pattern structure to be skipped. The cursor/marker advances to the next vector related statement.

4.2.5 Adjust Source Display

The **Adjust Source Display** performs the opposite function as the Adjust Waveform Display described above. In this case, the waveform cursor is first placed over a desired location in the waveform window. Then when the Adjust Source Display button is activated, the cursor in the source file window is adjusted to display the vector line which generated the waveform at the selected location.

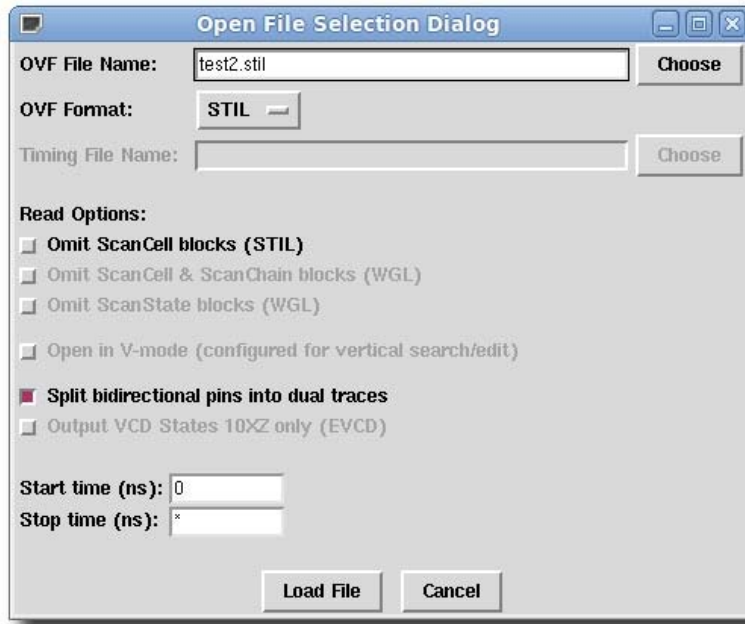
4.2.6 Terminate Session

The **Terminate Session** button is used to exit DFTView and the waveform tool. The corresponding windows will be closed, and intermediate files will be deleted.

4.3 Menu Options

4.3.1 File Menu

File→Open : this selection is used to load a new source file into the DFTView text window. It also causes the file to be mapped to its equivalent waveform in the waveform display window. During this operation the source file is checked for syntax errors and if any are found an error message is displayed, with a more detailed message in the parent window. This selection causes a query window to appear which has a text field for the source file name and, if necessary, a separate timing file name. The query window also has a Choose button which allows you to browse the files on your machine and select the desired one to be loaded. Once the source file is chosen, DFTView will try to determine the format of the file by looking at the file extension, and use this information to set the "OVF Format" selection. Always check that this selection is correct and make the selection manually if it is not correct.



The **File**→**Open** screen contains several optional Read parameters. These can be selected when reading STIL and WGL files to accomplish 2 things: minimize the size of the file to be displayed in the text widget, and to improve the waveform generation time when you do not need to look at the entire timeline. Loading and displaying very large files can take a significant amount of processing time and these options are provided to help you minimize these times.

It is often the case that Scan Cell name and list information takes up a significant percentage of STIL and especially WGL file sizes. For viewing these files and their corresponding waveforms, this information is usually not needed. Selecting the Omit options for these blocks may greatly reduce the amount of text data that needs to be brought-up in the text widget. In addition, for those cases where a user knows the particular area (range) in the test timeline that they would like to view, it can save substantial translation compute time by specifying that time range with the Start and Stop time parameters. The default for these is 0 to * (max time in file). You can also split bidirectional signals into separate input and output signals (this is enabled by default – see section 4.5) and for 93K format files you can enable V-mode (Section 5). The “Output VCD States 10XZ only” is only available when reading EVCD files. With this option enabled DFTView will display simplified waveforms using only the VCD states 1,0,X and Z.

Tip: When loading files which will generate large numbers of vectors (millions or more) you may find loading time to be longer than expected. DFTView flattens loops and repeats when generating the vectors so that they can be displayed, and this can take some time. It's highly recommended that you avail of the Start and Stop time parameters which allow you to quickly load an area of interest without having to load every vector.

Tip: *Gzipped input source and vector files will be unzipped automatically if a .gz extension is detected. Note that in this release, with formats that require one or more timing files, the vector file can be gzipped but the timing file(s) **must not**. If your timing files are gzipped you must manually unzip them first.*

File→Close : this selection will close the currently open and loaded file, and clear both the text and waveform windows. It is necessary to close the current file before you can open another one.

File→Regenerate Display : selection is used to update the data displayed in the waveform window, after modifying the source file vector or timing data with the DFTView editor. This selection does NOT update the original source file specified on the command line, the command file or specified with the File->Open menu. This selection functions identically to the Regenerate Display button.

4.3.2 Edit Menu

Edit→Go To Line Number: this selection brings up a query window with a text field for entering a line number. After entering a line number, activating the “Go To” button will cause the cursor in the DFTView text display window to move to the specified line number in the source file. Activating the “**Adjust Waveform Display**” button (Section 4.2.2) after this will then cause the waveform display to move to the corresponding location.

Edit→Go to Time Point: This selection brings up a query window with a text field for entering a time point (in nanoseconds). After entering a time point number, activating the “Go To” button will cause the cursor in the DFTView text display window to move to line in the source file most closely matched with the requested time point. Activating the “**Adjust Waveform Display**” button (Section 4.2.2) after this will then cause the waveform display to move to the corresponding location.

Edit→Search: this selection brings up a query window where a text string can be entered. The “Find” button in the window can then be used to position the cursor over the location in the DFTView text window where this text finds a match. After one search, the “Find Previous” and “Find Next” buttons can be used to continue the search.



4.3.3 Help Menu

The **Help** menu is located on the right-hand side of the menu bar. The first item is **About** which will display an application banner which will indicate which version of the DFTView you are running. (Note: If at any time you have a question for Source III please include the version number of the application, visible in this "About Box" or simply by typing `DFTView -v` on the command line). Besides the "About Box", the **Help** menu contains a series of entries each of which will display the application's built-in help for that option. So for example, selecting **Help**→**File**→**Save Source** will display the help text for the **File**→**Save Source** menu option.

4.4 Error Detection

If DFTView detects any error while parsing STIL or WGL format input files, it will not attempt to display any waveform data in the viewer. It will display the input source file in the DFTView text editor as usual and will then jump to the line in the input file which generated the error and highlight. In this example, the *Procedures* statement has been misspelled:

The screenshot shows the DFTView SourceFile: s21_broken.stil window. The text editor displays the following code:

```

"P CLK" { P { '0ns' D; '10ns' U; '30ns' D; } }
"default_Out_Timing" { X { '0ns' X; } }
"default_Out_Timing" { L { '0ns' X; '40ns' L; } }
"default_Out_Timing" { H { '0ns' X; '40ns' H; } }
"default_Out_Timing" { T { '0ns' X; '40ns' T; } }
// "default_Out_Timing" { XLHT { '0ns' X; '40ns' X/L/H/T; } }
}
}
PatternBurst "_burst_" {
  PatList {
    "_pattern_" ;
  }
}
PatternExec {
  // Timing "";
  PatternBurst "_burst_";
}
Procedures (
"load_unload" {
  W " default_WFT ";
  V { "P_CLK"=0; "P_CS1_N"=0; "P_CS2_N"=0; "P_IACK1_N"=0; "P_IACK2_N"=0; "P_MUSTANG_SE_N"=0;
    "P_ATE_N"=0; "P_RESET_N"=1; }
  Shift {
    W " default_WFT ";
    V { " _si"=#; " _so"=#; "P_CLK"=P; }
  }
}
"capture_P_CLK" {
  W " default_WFT ";
  "forcePI": V { " _po"=\r37 X ; " _pi"=\r55 # ; }
  "measureP0": V { " _po"=\r37 # ; }
  "pulse": V { " _po"=\r37 X ; "P_CLK"=P; }
}
"capture_P_CS1_N" {
  W " default_WFT ";
  "forcePI": V { " _po"=\r37 X ; " _pi"=\r55 # ; }
  "measureP0": V { " _po"=\r37 # ; }
}

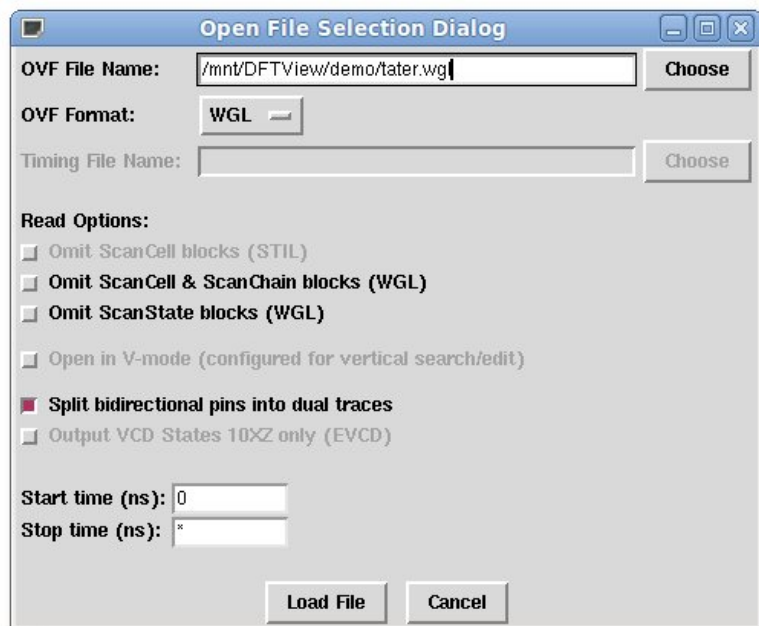
```

The word "Procedures" is highlighted in red, indicating a syntax error. The status bar at the bottom right shows "Edit mode: Normal". Below the text editor are several buttons: Regenerate Display, Adjust Waveform Display, Previous Vector, Next Vector, Adjust Source Display, and Terminate Session.

In addition to this, the relevant error message will be displayed in a standard dialog box. After you've dismissed the error box you can get it to display the error again by clicking on the little “stop” icon next to the highlighted error. At this point you can manually edit the file in DFTView and when you're ready click the “Regenerate Display” to re-parse the edited file. This will cause DFTView to re-parse the file, and will at this point display the waveforms in the viewer, or highlight additional error messages if detected.

4.5 Separating Bidirectional Pins

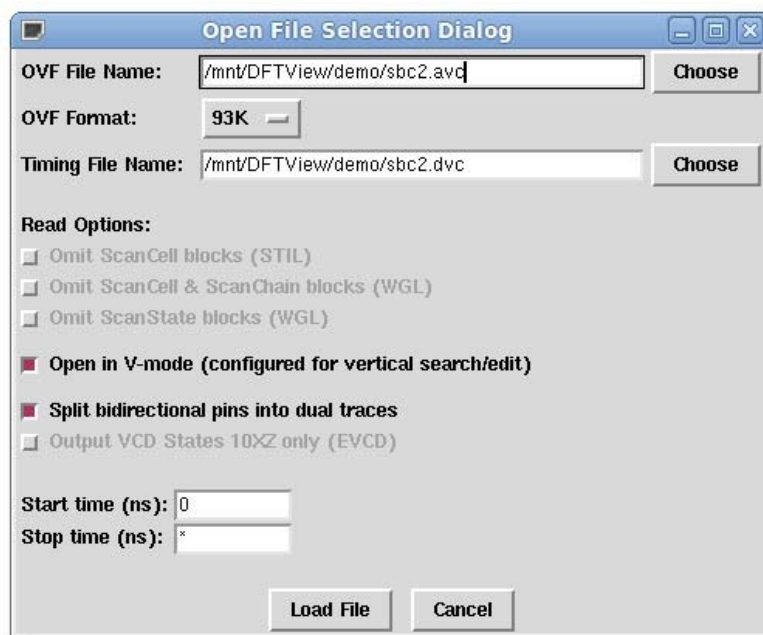
All supported input formats now support a bidirectional pin option, available in the File/Open dialog. When this option is checked all bidirectional pins in the input file are split into input and output pins, clearly labeled, in the resulting waveform display. This does not in any way alter the content of the input file. This option is only available via the **File/Open** dialog (Section 4.3.1). Note that this option is *always* enabled by default. Uncheck this box if you wish to view bidirectional signals as a single trace in the waveform viewer.



5 DFTView OPERATION: V-mode

5.1 Overview

When opening 93k format files in the File/Open dialog, the option to enable V-mode will be available (it will be disabled for other input formats). Selecting this option will alter how the input file is presented to you (although the various controls for interacting with the waveform viewer will not change). As described in “**DFTView Invocation**” (Section 3), if passing the names of the input vector and timing files on the command line you can also specify the `-vmode` command line argument to switch automatically into V-mode.



With V-mode enabled, pin state data is displayed in tabular fashion where each column represents the state of a single pin over time, and above the main display a secondary display will vertically label each pin name, aligned over the correct pin state column. Unlike the normal edit mode, lines in V-mode will not be wrapped automatically, and instead a horizontal scrollbar will appear to allow you to view additional pins if they do not all fit in the window at once.

When displayed in V-mode the file will be represented as a series of vectors and possibly loops (you can identify these by the SQPG statement delimiters). Other statements such as SCAN are not displayed, and if your input vector file contains SCAN statements these will be automatically flattened into a series of vectors before being displayed. If you alter and save the file note that it will **not** convert these flattened vectors back into SCAN statements.

5.2 Vertical Search

Along with the regular Search option (Section 4.3.2) V-mode also provides additional search behavior for the application. In normal mode, search is performed in a standard “text-editor” fashion, scanning line by line until a match is found. As seen in the extra **Edit** menu options above, both “Vertical Search” and “Vertical Search & Replace” options are available. The Vertical Search operation allows the user to search vertically on any selected column representing a single pin, although you can elect to search all columns as well. If all pins are being searched each pin will be highlighted in turn as its pin state data is searched for a matching pattern, and because this can be a time-consuming operation it can be interrupted at any time. A companion operation to this, the Vertical Search/Replace option where you can search for and automatically replace pin state patterns. This will only operate on the highlighted pin (column).



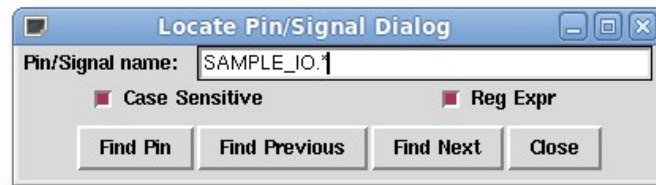
5.3 Vertical Search/Replace

In V-mode you can search for and replace a sequence of state characters on a single pin. With a pin selected you can call up this dialog box and use it to substitute new state sequences on the selected pin. Use the “Limit to vector range” entries if you wish to limit the “Replace All” operation to within the range you specify. As shown here, when you use the “Replace All” button, the number of substitutions performed will be indicated near the bottom right of the dialog box.



5.4 Locate Pin/Signal

This option allows you to quickly locate a pin or pingroup. It will pop up a dialog very similar to that which you've already seen for normal Search (Section 4.3.2):



You can search by absolute pin name, or use case sensitivity and/or regular expressions to widen your search. The search is always started from the left-most pin as displayed in DFTView. If a pin is located the vertical cursor will be moved immediately. Use the “Find Previous” and “Find Next” buttons to search left or right for additional pins matching your search parameters.

5.5 Smash and Compress

The **Edit** menu also provides additional options when a file has been opened in V-mode. If the cursor is sitting on a vector line which has a repeat value > 1, or is inside a loop block (wrapped in SQPG RPTV and SQPG PADDING statements) the “Smash” will be enabled and the arrow cursor to the left of the text pane will change to bright green. Such vectors and loops can be “smashed” (also known as “expanded” or “flattened”) such that each repeated vector is explicitly listed in the source window (similarly for loops where each set of vectors will be explicitly listed). For convenience, a smashed repeat vector or loop will be highlighted in light blue so they can be easily and quickly identified. The companion “Smash all Repeats” will iteratively smash any vector or loops with a repeat count greater than one. Note that for large files this process may take some time to complete. Complementing these “Smash” options the **Edit** menu also provides both a “Compress” and “Compress all Repeats” option.

The “Compress” operation will only be enabled when the cursor is sitting on a previously smashed block or loop. As with the “Smash” option, if the arrow cursor is aligned with such a previously smashed entry it will again change color to green indicating the “Compress” operation is available. The text-editor cursor may be placed anywhere in the smashed block, and selecting this option will replace the block with the original single vector line or loop block. Just as you'd expect, the “Compress all Repeats” option will compress all vectors or loops which have been previously smashed. The screenshot on the following page shows a smashed repeat vector.

Tip: When using the “Smash” and “Compress” operations it is possible for the cycle/vector correlation to go out of sync. If the cycle and vector numbers (Section 5.1) both show zeros or otherwise seem to be incorrect, and especially if the “Regenerate Display” button is flashing, don't forget to click the “Regenerate Display” button to refresh the waveform display and vector numbers (section 4.2.1).

5.7.1 File Format

This is accomplished by way of a pin formatting file which can be loaded into DFTView via the **Load V-mode format file** option in the **Edit** menu, and used to define which signals should be viewed, and in which order. To undo the operation, select **Remove V-mode formatting** from the **Edit** menu.

Tip: *If pin formatting is applied (i.e. you have already loaded a V-mode format file), you can re-apply this format file after editing it in an external editor, or apply a different format file, by using the same menu option which will now read "Load/reload V-mode format file". This will automatically remove the current pin formatting (preserving any changes to the vector file you may have made) and apply the updated pin formatting file instead.*

The pin formatting file is itself a very simple INI-style text file. It must at the very least contain a `[display]` section which tells DFTView which signals to display, e.g.:

```
[display]
INPUT[ 3 ]
INPUT[ 2 ]
INPUT[ 1 ]
INPUT[ 0 ]
SPACER
CLK_A
CLK_B
```

In this example, four input pins constituting a bus should be displayed, followed by a blank column and then two clock signals. The SPACER statement can be used as many times as you wish to further separate out signals you wish to view. These spaces will appear as blank columns in the vector display (text editor), but there will be no corresponding spaces in the waveform display. However the signals will be displayed in the same order as listed, both in the text editor and the waveform viewer. You may put comments in the format file by beginning the line with a # character, and blank lines and spaces are allowed. Section titles such as `[display]` are not case-sensitive, but note that pin/signal names are.

Changes made to any one of these pin subsets will be retained if the pin formatting is subsequently removed (via the **Remove V-mode Formatting** option in the **Edit** menu).

5.7.2 Named Pin Groups

You can also optionally define a pin group via a `[pingroup]` section. The first line in such a section should be the name of the pin group, and subsequent lines should list the signals making up this pin group, in the order you wish to add them. The name of a pin group must not be the same as the name of an individual signal or the pin group will be ignored. The signals can be listed on one line or split across multiple lines. The following two pin group sections are equivalent:

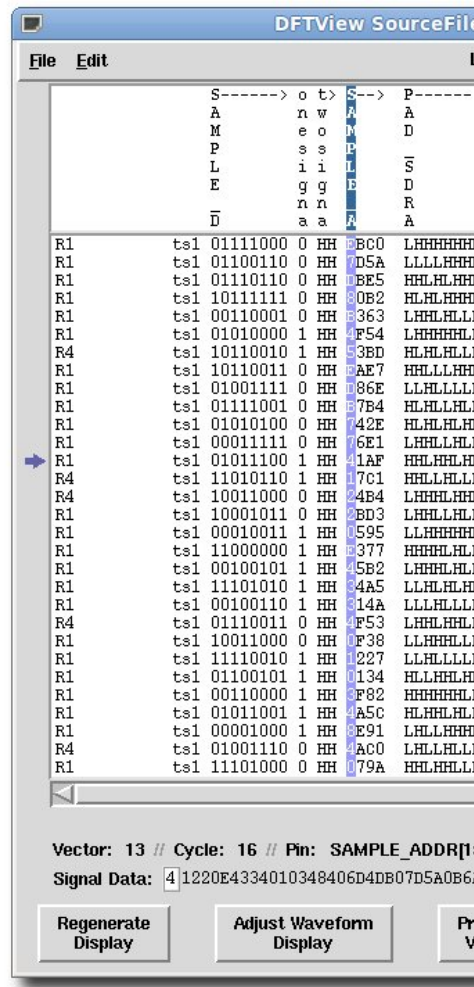
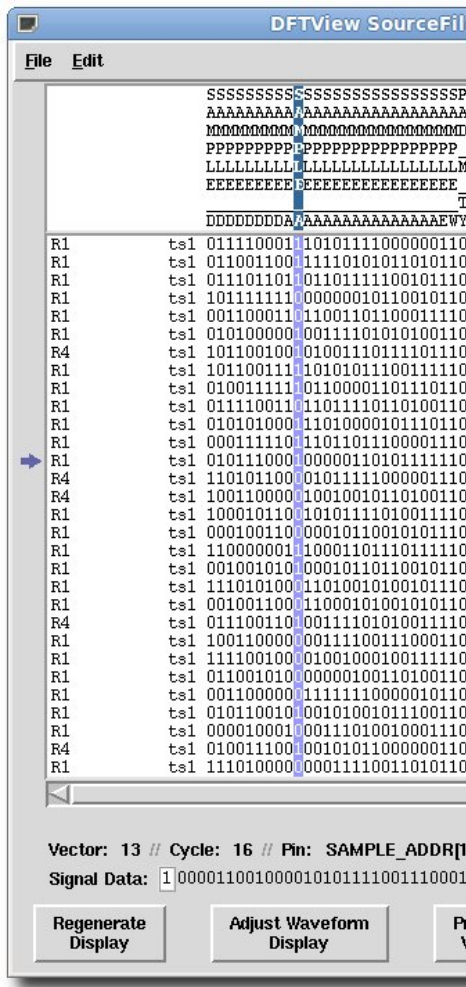
```
[pingroup]
name=INPUT[ 3:0 ]
signals=INPUT[ 3 ], INPUT[ 2 ], INPUT[ 1 ], INPUT[ 0 ]
```

```
[pingroup]
name=INPUT[3:0]
signals=INPUT[3],INPUT[2]
signals=INPUT[1]
signals=INPUT[0]
```

The name of the pin group can then be listed in the [display] section and that complete group will be displayed. You may *not* list a signal or pingroup twice in the [display] section, nor may you specify that a pin be displayed more than once, either alone or in any pingroups.. It is valid to define several pin groups without having them displayed. So using the example above, to hide the INPUT[3:0] pin group simply omit it from the [display] section.

5.7.3 Display Radix

Pin groups can be displayed in binary or hexadecimal. These side-by-side screenshot samples demonstrate a file displayed in DFTView with V-mode, before and after pin formatting has been applied. Notice how the hexadecimal display can make it easier to read the state of the selected bus (column).



By default a pin group will be displayed in binary form, and if no formatting option is listed after a signal or pin group in the `[display]` section, the BIN formatting option (case-insensitive) is assumed. When a pin group is displayed in BIN format the Vector/Cycle/Pin information near the bottom of the window will indicate both pingroup and component pin name at the cursor position where applicable.

You may also specify HEX in which case each four pins will be translated in order into the relevant hexadecimal character if possible.

```
[display]
SPACER
CLK
INPUT[3:0]    hex
```

For example, if at a given vector our input file for pins INPUT[3], INPUT[2], INPUT[1] and INPUT[0] listed the states as 0,1,0,0, then in hex that would be displayed as 4. Likewise for low/high pin states, in this case LHLL. If the four states cannot be combined into a hex character, a '?' character will be displayed instead. So four pin states 01XX or X00X would both be displayed as ? (but will correctly revert to their original states once pin formatting is removed). One exception is if all four characters are the same state, so XXXX would be displayed as X. Hexadecimal values may be edited and the change will be retained when pin formatting is removed. Thus changing a 0 to an F which displayed in hex will change the original pin states from 0000 to 1111 or LLLL to HHHH as applicable.

Finally, note that pin states are interpreted in the order they are listed. So to display a bus of four pins in most-significant-bit (MSB) order they should be listed low to high, while for LSB ordering they should be listed high to low. Thus in the above examples the INPUT bus is interpreted in LSB order.



Source III, Inc.
 3940 Park Drive, Ste. #20-342
 El Dorado Hills, CA 95762
 (916) 941-9403
www.sourceiii.com