Know More About Verilog-A Parser in SmartSpice

Introduction

In recent years, Verilog-AMS Hardware Description Language (Verilog-A) has been widely used in analog and mixed-signal design. Correspondingly, most EDA vendors provide simulation tools for Verilog-A. As one of those vendors, SIMUCAD also added support of Verilog-A in SmartSpice several years ago. Recently, some improvements have been made to the Verilog-A parser in SmartSpice, which can help to improve the performance of design simulation and project management. In this application note, those improvements will be introduced in detail, so users can understand better and take full advantage of the tool.

File Management Improvement

In SmartSpice, the Verilog-A parser needs to generate some files after processing the user's source code. Originally, those files were stored at the same location as the project file. The mix of input and output files could make it inconvenient for project management. To solve this issue, an improvement has been made in the SmartSpice Verilog-A parser. The improved Verilog-A parser creates a directory tree under the project file location and writes all resulting files to this directory. The directory tree will be created as shown in Fig. 1.

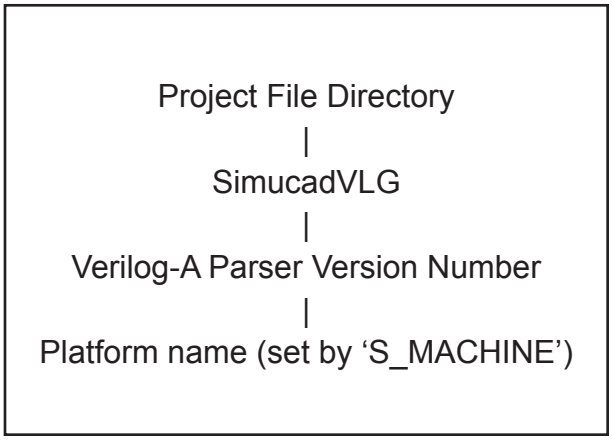

Fig. 1 Directory Tree Pattern.

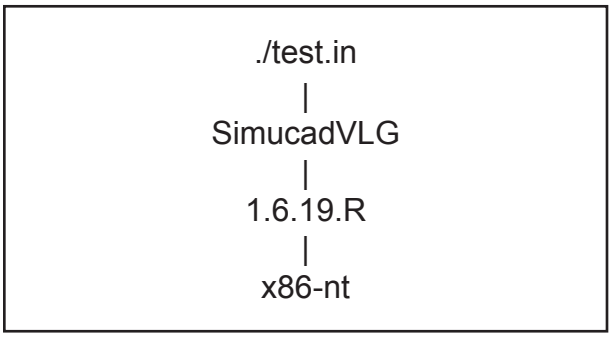

Fig. 2 Directory Example

For example, for project file=./test.in, Verilog-A Parser version=1.6.19.R, S_MACHINE=x86-nt, the Verilog-A parser will create a directory tree as shown in Fig. 2. With this directory tree, input and output files are separated, and it will be easy to manage the resulting files generated by the Verilog-A parser.

Reuse of Verilog-A Parser Results

When SmartSpice encounters a file coded in Verilog-A (in non-"SCI" mode), its Verilog-A parser processes the file and generates shared libraries that will be used by SmartSpice in simulation. Generating those shared libraries could be very time consuming. Processing an unchanged Verilog-A file in every simulation is unnecessary and wastes time. An enhancement has been made in SmartSpice where by shared libraries are now reusable. If a Verilog-A source file and its dependent files didn't change, the shared libraries generated before from this file can be reused without preprocessing by the parser. Therefore a Verilog-A file just needs to be processed once by the Verilog-A parser, while its results can be used repeatedly in SmartSpice shortening processing time.

From SmartSpice output messages, it is easy to know whether the Verilog-A parser is used to process a file or previous results are reused. For example, the following message:

(VERILOGA): Compiling file ..... 'D:\test_case\laplace.va'.

shows that file “laplace.va” is processing by the Verilog-A parser, while the message below

(VERILOGA): Using existing model(s) from ‘D:\test_case\laplace.va’.

tells that “lapace.va” doesn’t need to be processed by the parser because previous results are available for reuse.

Independent Verilog-A Parser

Previously, the Verilog-A parser was not an independent program. It was just a module of SmartSpice and could only be called by SmartSpice. Also, any new Verilog-A version would require reinstalling SmartSpice. In current SmartSpice, the Verilog-A parser is an independent program, which can be called by SmartSpice or run independently. Thus, one can run just the parser to process Verilog-A files without using SmartSpice, and the shared libraries generated by the parser can be directly used by SmartSpice in simulations. Also, when a new version of the Verilog-A parser is available, only the parser needs to be installed while SmartSpice remains unchanged, which makes maintenance easier. Furthermore, an independent Verilog-A parser can process multiple files with just one command line. For example, shell command:

```
> veriloga –l *.va –vcc
```

will process all Verilog-A files in the current working directory, which is both powerful and simple.

Several things are worth mentioning in running the Verilog-A parser from command line. First, among those options, “-l” is better to be set always since the log files generated by Verilog-A parser contain useful information, especially for error debugging. Second, on Windows, if the environment parameter “LIBVLG_PATH” isn’t set, the option “-libvlgpath” must be used to specify the path where the Windows C compiler can get libVLG library correctly. Third, option “-f” is used to set the file name displayed in all messages replacing the input file name. For example, command

```
>veriloga –l .\test\example1\example.va
–f example.va –cc
```

will use “example.va” to replace “.\test\example1\example.va” in every message displayed. This option can shorten message length and make it easy to read. However, this option can not be used if the command line contains multiple input files.

Like SmartSpice, now users can run specific version of the Verilog-A parser by using “-V” option in the command line. For example, shell command

```
>veriloga –V 1.6.17.R *.va –vcc
```

will use Verilog-A parser version 1.6.17.R to process files. If the version specified is not installed, the following message will appear:

```
“Command line –V “1.6.17.R” not found.
    Exiting.
```

Available versions for this platform are: 1.6.15.R and 1.6.13.R.

When processing Verilog-A files from SmartSpice, this message will also appear in the SmartSpice output window if the libVLG library specified in the SmartSpice ModelLib configuration file doesn’t have a corresponding version of the Verilog-A parser installed. The version match requirement is important. SmartSpice requires that the loaded libVLG library version matches the Verilog-A parser version. The version of libVLG library is specified in the ModelLib configuration file. Generally it is not an issue since SmartSpice guarantees that the correct version of the Verilog-A parser will be used. However, if a user wants to use the Verilog-A parser independently to process Verilog-A files first, and later reuse those generated results directly in SmartSpice, then one must make sure that the version of the Verilog-A parser used doing the preprocessing matches the libVLG library version used by SmartSpice. Otherwise, SmartSpice will call the correct Verilog-A version to process those Verilog-A files, and the advantage of reusing Verilog-A parser results will be lost.

Conclusion

In this note, some recent changes made on the Verilog-A parser of SmartSpice were introduced. Those improvements make project management easier, improve simulation performance, and give users more flexibility. Understanding those improvements will help users to make better use of the tool and complete their job more efficiently.