Abstract

One big bottleneck for VHDL-based system simulation was the weak performance of the VHDL simulators. The new approach of the Native-Compiled-Code based simulators is very promising. The main objectives of the following evaluations were performance and compliance to VHDL-1076. The test cases (in total 80000 lines of code) of the benchmarking were written by hardware designers with different style of coding. More than 400 single results delivered precise information on the software simulation systems of 5 different vendors and one hardware accelerator.

1 Introduction

In the past, most of the CAE-vendors struggled with supporting full VHDL-1076 by their VHDL simulators. Since most of the key players do now support full VHDL, the effort is spent in making VHDL simulation possible for large projects and applicable to system simulation.

One big bottleneck for VHDL-based system simulation was the weak performance of the VHDL simulators compared to traditional gate-level simulators. Now one can observe that the CAE-vendors are attacking this challenge. Nearly every CAE-vendor claims to have the fastest simulator. The new approach of the Native-Compiled-Code based simulators is very promising. New releases of simulators and new products recently appeared made an evaluation necessary in order to get the correct information of the key simulator products, because information from vendors is often doubtful.

Benchmarking seems to be a hot topic currently, on the last conferences we have had interesting panels about it. In the last time we could also see a few small consulting companies coming up with the service of doing benchmarking. But with third party or vendor benchmarks were arises the problem that they do not have real applications as test cases. Their models seem to be very artificial, because it is to expensive to create huge test cases especially for benchmarks. Users also believe that vendor benchmarks are tailored to their own systems.

In this paper the reader gets valuable information on performance issues of VHDL simulators. Since the benchmarking is done in batch mode and fully automatically, it can be repeated in a short time, if for instance, a new version of a simulator is released.

2 Used Simulators

Most of the models used as benchmarks were developed using the SYNOPSYS simulator VSS V2.2. The benchmarking was extended to the simulators Optium from VANTAGE, to the SUN-versions of V-System from Model Technology, Voyager from IKOS and Leapfrog from CADENCE.

The simulator QuickVHDL offered by MENTOR is identical to the simulator from Model Technology.

Recently the benchmarking was extended to the simulation system Paradigm ViP V1.2-beta from ZYCAD. This system is actually a hardware accelerator for VHDL.

3 Different Simulator Approaches

Today we differentiate four categories of simulators on the market[1]:

- **Interpreter**
  The VHDL-Code is compiled to an internal pseudo-code, which is interpreted by the simulator kernel. These systems provide very fast compilation but require a relatively long simulation time. Simulators of this kind are VSS from SYNOPSYS up to V3.0 and V-System from Model Technology up to V2.5.
• **C-Code based simulators**
These systems generate C-Code from the original VHDL-Code. The C-Code is then compiled to object code of the host platform by a normal C-Compiler. An executable model is built by linking the resulting object code together with object code for the simulator kernel. This method improves simulation time but this happens at the expense of compilation time due to the slow process of generating and compiling C-Code. Simulators of this kind are the simulators Optium from VANTAGE, V3.1 from SYNOPSYS, VHDL-2000 from RACAL REDAC, Voyager from IKOS and VLK(Scout) from CLSI(COMPASS).

• **Native-Compiled-Code based simulators**
Native-Compiled-Code based simulators directly compile VHDL to the native code of the microprocessor of the host platform. Due to the absence of the time-consuming C-Code generation this approach combines the advantages of the two solutions mentioned above, short compilation time and very fast simulation. It raises, however, the issue of the software portability. The maintenance effort for supporting different host platforms is much higher than for the two other approaches.

The Native-Compiled-Code based simulators are on the market since Q3/93. The products are Leapfrog from CADENCE, the V-System since V4.1 from Model Technology, LogicBench from IBM and QuickVHDL from MENTOR, which is identical to V-System from Model Technology.¹

• **Hardware Accelerator**
There is currently only one system, the Paradigm ViP from ZYCAD, on the market. This system consists of a special hardware, the ViP-Box executing the simulation, and software for compiling and elaborating the VHDL source code on a host workstation, which is connected to the ViP-Box by a special S-Bus card. The elaborated design is loaded on the ViP and executed on special processors inside the ViP.

4 Environmental Conditions

4.1 VHDL Models used for the Benchmarking

Most of the models used as benchmarks were developed using the SYNOPSYS simulator VSS V2.2. Therefore the logic type mainly used is MVL7.

The models have altogether about 80000 Lines of Code (LOC). The models were written by different hardware designers with different style of coding. In contrast to many test cases of vendors or third parties these models represent real design examples. In comparison to already published benchmark results [2], [3] it looks that our testbed is the most comprehensive one, so that the results are a relatively objective criterion for making decisions. Table 1 gives a short overview of the test cases run for the benchmarking.

<table>
<thead>
<tr>
<th>Model</th>
<th>LOC</th>
<th>Transactions</th>
<th>Characteristics of the model</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSK</td>
<td>4434</td>
<td>12269</td>
<td>Many small models</td>
</tr>
<tr>
<td>8051</td>
<td>6302</td>
<td>43761897</td>
<td>System simulation</td>
</tr>
<tr>
<td>LCD</td>
<td>1986</td>
<td>20356</td>
<td>LCD-Display</td>
</tr>
<tr>
<td>BADGE</td>
<td>12487</td>
<td>3936787</td>
<td>System simulation</td>
</tr>
<tr>
<td>DIVI</td>
<td>984</td>
<td>4521104</td>
<td>Divider</td>
</tr>
<tr>
<td>INV1000</td>
<td>2011</td>
<td>504504</td>
<td>Generated model</td>
</tr>
<tr>
<td>KOM</td>
<td>41160</td>
<td>35600000</td>
<td>Telecom ASIC</td>
</tr>
<tr>
<td>SIG</td>
<td>541</td>
<td>0</td>
<td>Pure sequential code</td>
</tr>
<tr>
<td>SPARC</td>
<td>1913</td>
<td>2059939</td>
<td>Sparc processor</td>
</tr>
</tbody>
</table>

Table 1: Overview of the test cases

The models were packed to the following groups:

• **VSK**
This group comprises a lot of small models (about 60 models), which are normally used for VHDL training. The simulation time is small, these models give however a good indication whether the simulators are compliant to VHDL-1076.

• **8051**
This is a relatively complex VHDL-model of a small system, a timer unit. The main part consists of a behavioral model of an 8051-microcontroller. The software of this system is loaded

¹Actually MENTOR acquired Model Technology in ’94
as assembler code in a memory device, that is also modeled in VHDL.

- **LCD**
  This is a pure behavioral model of an LCD-Display embedded in a test bench for simulation. The output of the LCD is done via TEXTIO.

- **BADGE**
  This is a system simulation of a board consisting of an FPGA, a RAM and the above mentioned LCD. About 50% is behavioral code for the LCD and the RAM, the other parts are written in a subset of a high-level synthesis tool.

- **DIVI**
  Parameterized VHDL model of a divider, written in the subset of the SYNOPSYS Design Compiler at RT level. The model generates a lot of assertion messages during simulation.

- **INV1000**
  Very simple generated model of 1000 chained inverters.

- **KOM**
  This is the largest and most complex one of the test cases used. The model specifies a telecommunication ASIC of 60 K gates. The major part of the model consists of synthesizable RT level code for the SYNOPSYS Design Compiler. The rest of it consists of behavioral code for some static RAMs and stimuli for the test bench.

- **SIG**
  Pure sequential code for testing of a new synthesis package called ssig_2 (subprogram ID 21 and ID 23). This is a very special model, since no signals are used in it. That means no transactions or events occur during simulation.

- **SPARC**
  Behavioral model of a Sparc processor. The simulation covered 10 µs of a machine program loaded into a ROM also modeled in VHDL.

### 5 Results

Table 2 to Table 4 give an overview to the results. The values in Table 2 and Table 3 are normalized on the fastest software simulation system. That means that a value of 5 indicates that this system needed 5 times more CPU-time than the fastest system.

The interpreter version from SYNOPSYS and the V-System from Model Technology provide the fastest analyzer. The analyzer of CADENCE's Leapfrog however is not far behind. Only the analyzer from the IKOS system and VANTAGE system used in the average 20 times (17 times) more CPU-time. On a sparc2 we found values between 0.5 LOC/sec/Mip (IKOS-Analyzer) and 38 LOC/sec/Mip (MTI-Analyzer). That means the overall compile-time for 100k LOC varies from 6 minutes to 120 minutes.

In the past some companies stressed their short analyze-time, but in general the analyze-time is not the bottleneck, because even in case of a very big design with long compilation time, the design is split up into a lot of small design units, which can be compiled separately due to the excellent module concept of VHDL. If the source code must be changed inside one design unit, only small portions of the complete design must be recompiled. This re-compilation is supported in nearly every system by the use of make-mechanisms.

Table 3 shows that Leapfrog from CADENCE clearly provided in all test cases the fastest software simulation by competitive compilation time. Only in three cases we can see a speed-up by a factor of three in using the ViP-Box, which is a little bit disappointing if we look to the former announcements.

Model Technology's move from an interpreter solution to a native-compiled-code based simulator results in a speed-up of 3. SYNOPSYS offers with its version V3.1 an interpreter with the same performance as version V3.0 and a new compiled-code based simulator with a performance improvement with a factor of three by the disadvantage of a higher compile-time. The VANTAGE simulator is with its values somewhere in between (in the average about 4 times slower than Leapfrog).

### 4.2 Hardware

The Unix platform used in the benchmarks was a SUN Sparcstation2 with 32 MByte RAM, 96 MByte swap space under the operating system SUN OS 4.1.3. The processor of this machine has a performance rate of 28 Mips. All files affected by the simulation were located on the hard disc belonging to this machine.

Some simulation runs were repeated on a SUN Sparcstation10 (96 Mips) with 32 MByte RAM and 96 MByte swap space. Later on the benchmarking was redone on a SUN Sparcstation20.

In case of the Paradigm ViP from ZYCAD, the ViP-Box was a two board machine (Paradigm 2004), which consists of 14 special processors for executing the simulation.
<table>
<thead>
<tr>
<th></th>
<th>SYNOPSYS</th>
<th>SYNOPSYS</th>
<th>IKOS</th>
<th>Model Tech</th>
<th>VANTAGE</th>
<th>CADENCE</th>
<th>ZYCAD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>V3.2</td>
<td>V3.0a</td>
<td>V1.41</td>
<td>V4.2</td>
<td>V5.0</td>
<td>Leapfrog 2.1</td>
<td>ViP 2.1</td>
</tr>
<tr>
<td>VSK</td>
<td>3.1</td>
<td>1.4</td>
<td>28.8</td>
<td>1.0</td>
<td>14.7</td>
<td>3.4</td>
<td>10.3</td>
</tr>
<tr>
<td>8051</td>
<td>3.8</td>
<td>1.9</td>
<td>19.3</td>
<td>1.0</td>
<td>21.9</td>
<td>4.1</td>
<td>15.2</td>
</tr>
<tr>
<td>LCD</td>
<td>2.0</td>
<td>1.0</td>
<td>13.9</td>
<td>1.3</td>
<td>9.7</td>
<td>1.6</td>
<td>8.9</td>
</tr>
<tr>
<td>BADGE</td>
<td>1.8</td>
<td>1.0</td>
<td>10.2</td>
<td>1.0</td>
<td>12.3</td>
<td>1.5</td>
<td>8.3</td>
</tr>
<tr>
<td>DIVI</td>
<td>1.7</td>
<td>1.0</td>
<td>15.7</td>
<td>2.5</td>
<td>8.8</td>
<td>2.5</td>
<td>15.6</td>
</tr>
<tr>
<td>INV1000</td>
<td>3.9</td>
<td>1.8</td>
<td>27.0</td>
<td>1.0</td>
<td>43.3</td>
<td>8.7</td>
<td>11.0</td>
</tr>
<tr>
<td>KOM</td>
<td>1.8</td>
<td>1.2</td>
<td>15.9</td>
<td>1.0</td>
<td>8.3</td>
<td>2.1</td>
<td>9.1</td>
</tr>
<tr>
<td>SIG</td>
<td>4.0</td>
<td>2.0</td>
<td>-</td>
<td>1.0</td>
<td>17.0</td>
<td>3.0</td>
<td>15.0</td>
</tr>
<tr>
<td>SPARC</td>
<td>2.7</td>
<td>1.6</td>
<td>29.5</td>
<td>1.0</td>
<td>13.8</td>
<td>6.7</td>
<td>3.5</td>
</tr>
<tr>
<td>Δ</td>
<td>2.7</td>
<td>1.4</td>
<td>20</td>
<td>1.0</td>
<td>17</td>
<td>3.5</td>
<td>10.8</td>
</tr>
</tbody>
</table>

Table 2: Comparison of the Analyze time (relative to the fastest Analyzer)

<table>
<thead>
<tr>
<th></th>
<th>SYNOPSYS</th>
<th>SYNOPSYS</th>
<th>IKOS</th>
<th>Model Tech</th>
<th>VANTAGE</th>
<th>CADENCE</th>
<th>ZYCAD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>V3.2</td>
<td>V3.0a</td>
<td>V1.41</td>
<td>V4.2</td>
<td>V5.0</td>
<td>Leapfrog 2.1</td>
<td>ViP 2.1</td>
</tr>
<tr>
<td>VSK</td>
<td>2.1</td>
<td>1.3</td>
<td>1.0</td>
<td>2.0</td>
<td>1.1</td>
<td>1.0</td>
<td>-</td>
</tr>
<tr>
<td>8051</td>
<td>-</td>
<td>4.4</td>
<td>3.9</td>
<td>2.5</td>
<td>4.6</td>
<td>1.0</td>
<td>-</td>
</tr>
<tr>
<td>LCD</td>
<td>1.1</td>
<td>3.5</td>
<td>2.7</td>
<td>1.7</td>
<td>3.1</td>
<td>1.0</td>
<td>0.3</td>
</tr>
<tr>
<td>BADGE</td>
<td>-</td>
<td>8.0</td>
<td>16.9</td>
<td>4.5</td>
<td>3.6</td>
<td>1.0</td>
<td>-</td>
</tr>
<tr>
<td>DIVI</td>
<td>6.3</td>
<td>17.9</td>
<td>8.1</td>
<td>5.7</td>
<td>7.0</td>
<td>1.0</td>
<td>4.8</td>
</tr>
<tr>
<td>INV1000</td>
<td>2.7</td>
<td>4.3</td>
<td>1.7</td>
<td>1.3</td>
<td>1.6</td>
<td>1.0</td>
<td>0.3</td>
</tr>
<tr>
<td>KOM</td>
<td>4.6</td>
<td>12.3</td>
<td>6.0</td>
<td>4.8</td>
<td>4.6</td>
<td>1.0</td>
<td>-</td>
</tr>
<tr>
<td>SIG</td>
<td>3.6</td>
<td>31.7</td>
<td>-</td>
<td>2.6</td>
<td>5.2</td>
<td>1.0</td>
<td>0.4</td>
</tr>
<tr>
<td>SPARC</td>
<td>1.0</td>
<td>11.8</td>
<td>1.8</td>
<td>2.2</td>
<td>3.3</td>
<td>1.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Δ</td>
<td>3</td>
<td>10</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>0.3</td>
</tr>
</tbody>
</table>

- Model could not be simulated

Table 3: Comparison of the simulation time (relative to the fastest software simulator)
Table 4: Performance of the Simulation in Transactions/sec/MIP on Sparstation2 (28 Mips)

5.1 Remarks to some Results

If we analyze the results given in the tables we can detect differences in the relative performance from one system to another depending on the circuit used for benchmarking. This makes evident that users should always be careful not to overemphasize the results from one single benchmark.

The relative compilation times do not vary strongly, but the transactions/sec/Mip reached by a system depends on the simulated circuit. Nevertheless, a ranking in terms of simulation performance would give similar results in almost all cases. This is true even if we look at some exotic test cases.

Some experiences were made with the SPARC model, in which a large RAM as well as a ROM was modeled originally using global signals. None of the analyzers used was able to compile this version of the model due to memory problems on the host. In a first modification of the VHDL code the ROM was modeled with variables. This version could only be analyzed and simulated by Leapfrog, however with a bad performance. In a third version also the model of the RAM was changed to a model using variables. Only this version could be used for the comparison of the simulators. The results listed in the tables belong to the third version.

The INV1000 model reaches a high performance peak. The reasons for that are the simple VHDL constructs used in that model. The model does not use any complicated user-defined data type, there is no resolution function called and the model does not read or write any file during simulation.

The models of the unit VSK are also special ones, because we have a bunch of very small models with single simulation time in parts of seconds. This can produce rounding errors, which can be accumulated during the simulation of this unit. But these models give a good indication about the VHDL-1076 compliance.

Closest to the reality of a user model are the large models like BADGE, DIVI and KOM. These models consist of a mix of behavioral code and RT-level code, which can be synthesized by commercial synthesis tools. The models did not produce a lot of output during simulation, i.e. only a few signals were traced during the simulation. If a lot of output has to be handled, all simulators used dramatically more CPU-time. Some of the simulators seem to be more optimized for that than others. Leapfrog did perform extremely good in simulating the DIVI-model. This model produces thousands of assertion messages.

5.2 Remarks to the ViP

If we look to the result tables, we see that not all of the models could be simulated by ViP. The reason for that is, that the current version of the system does only support a subset of VHDL. At a first glance to the not supported features of the language, it did not look bad. But during the evaluation a different experience was made. At the beginning only a small part of the VHDL code could be compiled. For the rest of the models, it was tried to rewrite the models. It turned out that this is a very time consuming and in some cases not practical task to do.²

The evaluation clearly showed that the compilation and elaboration time, a part which is done on the host workstation, is very high. As a consequence the turn around time of the ViP is rather bad and the main application will therefor be to run huge stimuli sets with simulation times of hours or maybe days. In some cases we found that Leapfrog from CADENCE was faster also in executing the simulation than ViP.

5.3 Influence of Different Options

During the benchmarking it was tried to use the simulators under the same conditions. Some tools offer the possibility of using special options to speed

²At the beginning of '95 ZYCAD announced that it stopped the further development of the ViP.
up the simulation under some circumstances. Range checking is very time consuming, also providing debug information and in general a lot of I/O-operations slow down the simulation. Turning off range checking, simulation without any debug information etc. results in an improvement of the simulation time for VANTAGE (from 10% to 50% in special cases), for CADENCE Leapfrog (from 5% to 30% in special cases). For the IKOS Voyager a tremendous speed up with a factor of four could be found in certain cases.

Vantage offers an Analyzer, which can distribute the compile process over a workstation ring. This can improve compilation time, i.e. for the test case KOM the compilation time (in this case elapsed time) could be reduced from 45 minutes to 15 minutes by using VCC (VANTAGE Concurrent Compiler) on a workstation ring consisting of 7 machines.

5.4 Comparison SUN Sparcstation2 with Sparcstation10 and Sparcstation20

The simulation runs were repeated on a SUN Sparcstation10 with 32 MByte RAM and 96 MByte swap space. This platform has a performance rate of 96 Mips (more than three times of a Sparcstation2). Nevertheless our evaluations showed only a speed up of two and a speed up around of three in case that a Sparcstation20 was used. This dearly indicates that all benchmarks must be done on the same hardware to do a fair comparison of the results.

6 Future Work

Since the benchmarking can be done fully automatically in batch mode, the results will be updated whenever a new version of a simulator is released.

In the near future the test cases will be extended to models which incorporate net lists, in which the basic elements are modeled with the soon coming up VITAL-Standard (VHDL-1076.4) for modeling library elements. Some vendors have already announced VHDL Gate-level-simulators, which can speed up net list simulation in case that VITAL primitives are used in the net lists.

It would also be very interesting to check the speedup of SpeedWave/MT a Multithreading Simulator from VANTAGE which was just released. This simulator can distribute the simulation tasks to the different processors available on a multi processor SUN workstation.

7 Conclusion

The main objectives of the evaluations were performance and compliance to VHDL-1076. The test cases (in total 80000 LOC) of the benchmarking were written by hardware designers with different style of coding. More than 400 single results delivered precise information on the evaluated simulation systems. The benchmarking showed that performance improvements in VHDL simulation have been made [4], [3], [5] in the past two years. Especially the new approach of the native-compiled-code based simulators fulfilled their promise of better simulation performance. But there is still a need for further improvement, because the ratio between simulation time and real time is in the order of $10^3 - 10^5$ depending on the circuit and the used simulator. That means if we want to simulate 1 sec of a system consisting of several ASICs, we need a simulation time of about 10 days.

Compliance of the simulation systems to VHDL-1076 seems to be no more a problem (without the ViP-Box). Only minor differences from one system to another causing some problems were found; they could be solved in a short time. Furthermore, the evaluation proved that the tools of the key players have reached more or less the same level of features and comfort of the user interfaces and debug facilities.

References