Automatic Checking of Students' Designs Using Built-In Selftest Methods

Norman Hendrich

Dept. of Computer Science, University of Hamburg
Vogt-Koelln-Str 30, D 22 527 Hamburg
Email: hendrich@informatik.uni-hamburg.de

Abstract While distance learning is attractive for many reasons, it also incurs some specific problems. One of them is the lack of direct interaction between teaching staff and the students, another one the large number of students for some courses. Therefore, the checking and correction of the homework submitted by the students often takes a lot of time and effort. This is especially true for digital logic design exercises, where a variety of functionally equivalent designs is possible.

In this paper we show how to apply the signature analysis technique based on linear-feedback shift-registers, well known from built-in selftest (BIST) research, to the automatic checking of the circuits designed by the students. As the selftest is run on the students' own PC or workstation, the system provides immediate feedback about whether a circuit is working, and it allows to reduce the number of false solutions submitted by the students.

1 WebAssign and Hades

Several management systems have been proposed to ease the distribution and correction of exercises for distance learning. One of these is the WebAssign system [1], built since 1997 in a cooperation of several German universities. Among the functions provided by WebAssign is a module for automatic checking of students' homework solutions, which are represented as HTML documents. This module currently allows to check (and correct) numerical values and the answers to simple yes-no or multiple-choice questions. Unfortunately, none of these functions is even remotely sufficient to check or rate the solutions to digital design assignments.

We therefore tried to integrate a means to automate the checking of digital logic homework assignments as far as possible. While several digital simulators suited to teaching are available (e.g. Diglog [2]), we selected Hades [3], a Java-based framework for digital system simulation and visualization. First, the Hades system is available free of charge for eductional use. Second, the software is very portable and runs on Windows, Linux, Solaris, Macintosh systems, and it can also be used as an applet. Third, the simulator provides all the simulation models required, including a complete set of gate-level models, and the linear-feedback shift registers and BILBO registers [4] used for running the selftests. Fourth, the networking functionality provided by Java allows the easy integration into the WebAssign system, which itself is also implemented in Java.

2 Built-In Selftest

The basic idea of built-in selftest methods is to increase the testability and to reduce the cost of testing of integrated circuits. A variety of algorithms and architectures have been

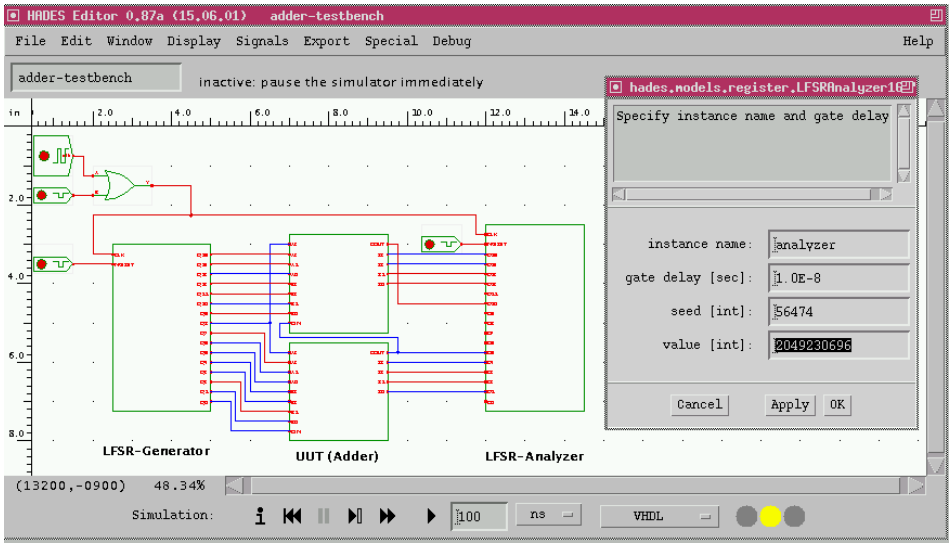

Figure 1: Signature-analysis example: Left: A schematic in the Hades editor showing the testbench circuit with LFSR-generator, the unit under test, and LFSR-signature analyzer. The dialog window shows the final signature value collected by the analysis register (highlighted). The simulation can also be run in batch-mode without the user-interface.

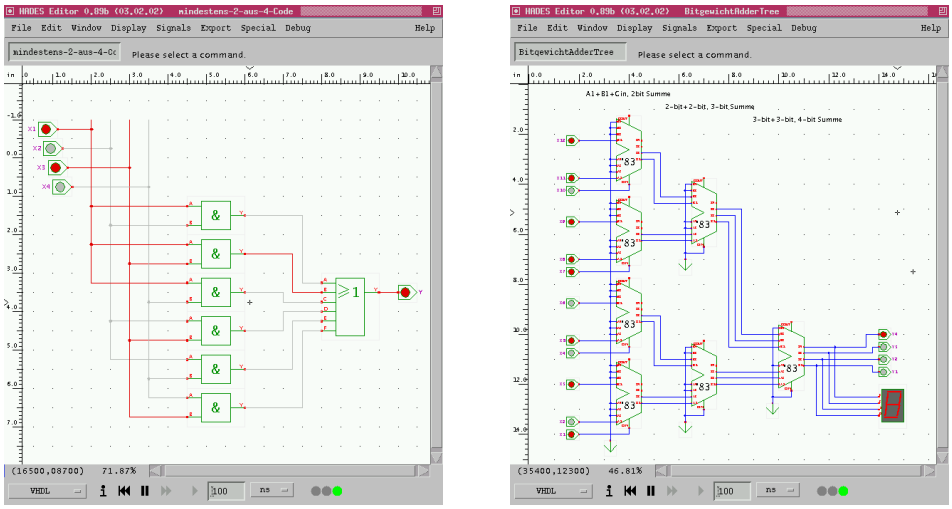

Figure 2: Two typical homework assignment design examples. Simple 2-out-of-4 decoder circuit (left) and a tree of adders to calculate the bit weight of the inputs (right). The template designs available to the students only include the I/O-components, into which the logic "has to be filled in". To comply with the presentation in the lectures, the German DIN symbols are used for the basic logic gates.

```
public int check_signature( int seed1, int seed2 ) {
  editor.loadDesign( "testbenches/bitcount8.hds" )
  analyzer = editor.getComponent( "LFSR-analyzer" );
  analyzer.setSeed( seed1 )
  generator = design.getComponent( "LFSR-generator" );
  generator.setSeed( seed2 );
  editor.runSimulation( time )
  signature = analyzer.getValue()
  return signature;
}
```

Figure 3: Pseudocode for a personalized signature check. The correct "master" signature can either be calculated or is found by running the simulation on a reference circuit.

proposed to this end (for an introduction see [5]). One of the most successful techniques is the use of linear feedback shift-registers for both pattern generation and signature analysis. From a hardware point of view, the main advantage of the LFSR-registers and the BILBO-registers in particular is the low cost, as the selftest logic can usually be integrated into the registers required for the normal system function. Also, LFSR-registers can be run at very high clock-rates which enables a full-speed test of the target logic. This in turn allows to run many test patterns and results in high fault-detection probabilities.

In the context of WebAssign, we use the signature analysis technique for other reasons. Most importantly, it is very simple to write or even auto-generate a testbench circuit based on LFSRs. Also, there is no need to write specific input stimuli for each assignment. Instead, the LFSR generators and signature analysis registers are initialized with suitable seed values and the simulation is run for a predefined number of clock cycles (see figure 3). Once the simulation is finished, the signature values are read out from the analysis registers and compared with "golden" values from a correct circuit. For small circuits it is also practical to run a complete simulation through all combinations of input values, if required.

As an example, the testbench design used for the selftest of a simple adder circuit is shown in figure 1. It consists of the LFSR-based generator register on the left, the LFSR-based signature analysis register on the right. The generator and signature registers are connected to the unit-under-test circuit, as well as clock and reset signals. In order to guarantee correct interfaces for the circuits, both the testbench design and an empty (interface only) circuit are provided to the students, who then have to design the missing adder logic.

Two examples of typical homework assignments are shown in figure 2. While the circuits are quite simple, it is easy to introduce errors, which can be hard to detect without the help of the simulator and the selftest.

2.1 Personalized Signatures

After designing the circuit, the student runs the selftest on the testbench, generating a first signature which is checked against a master signature provided together with the testbench. Only when both signatures match is the circuit working correctly. Note that Hades allows both batch-mode simulation and interactive simulation, where the circuit can be edited and corrected without leaving or restarting the editor.

Once the circuit is working, the student runs a second selftest with new start values for the LFSR registers, generating a second signature. Finally, the student submits the solution (that is, the Hades design files) together with the signature values to the WebAssign server. By using different initial seed values for the LFSR pattern generator and signature registers, we are able to personalize the exercises for each student.

For a simple circuit like the adder presented above, the main advantage of the selftest is the immediate feedback to the student whether the design is already correct. However, for larger designs and especially for sequential circuits with complex state machines, where a variety of solutions is possible, the selftest also greatly reduces the work required to check the students' solutions (and it still provides feedback to the student).

2.2 Timing Checks

Because Hades uses a discrete-event based simulation engine, the simulation includes gate-delays and allows timing checks on the circuits. For example, one of the homework assignments asks the students to design a carry-select adder from given 8-bit adder and multiplexer blocks. Therefore, the correct solution to the exercise does only differ in the timing, but not the functional behaviour, from a much simpler ripple-carry architecture. However, by running the selftest at a clock-speed that exposes the gate-delays, we can easily detect the slower (and wrong) design, without the need for additional timing checks or intricate waveform comparisons.

3 Status and Experiences

The system is currently in the final implementation and test phase, and it is scheduled for the summer semester courses [7]. While the server-side software is stable, the robust installation of Hades on the student's computers is still a concern, as many combinatons of operating system and Java virtual machines have to be supported. Initial student feedback should be available soon.

References

1. Brunsmann, J., Homrighausen, A., Six, H.-W., Voss, J., Assignments in a Virtual University - The WebAssign-System, *Proc. 19th World Conference on Open Learning and Distance Education*, Vienna, 1999.
2. Gillespie, D., and Lazzaro, J., *DigLOG and AnaLOG, Caltech VLSI CAD Tools*, California Institute of Technology, 1996 ftp://ftp.pcmp.caltech.edu/pub/chipmunk/
3. Hendrich, N., A Java-based framework for simulation and teaching: Hades, in *Microelectronis Education, Proc. EWME-2000*, 285-288, 2000
   tech-www.informatik.uni-hamburg.de/applets/hades
4. Koenemann, B., Mucha, J., and Zwiehoff, G., Built-in Logic Block Observation Techniques (BILBO), Proc. International Test Conference, 37-41, 1979
5. Lala, P.K., *Digital circuit testing and testability*, Academic press, 1997
6. Schiffmann, W., and Schmitz, R., *Technische Informatik I*, Springer 1999
7. Schiffmann, W., *Technische Informatik I*, lecture and assignments, Fernuniversität Hagen, 2002