Examples of Hardware Verification Using VIS

Abhijit Jas, Alper Sen, Anand Ramachandran, Cagdas Akturan, ChiaBin Liu
Debaleena Das, I-Min Liu, Jayanta Bhadra, Justin R. Denison, Kaustubh Das
Malay K. Ganai, Padmini Gopalakrishnan, Parminder S. Chhabra
Praveen K. Jaini, Rajat Chaudhry, Ram Narayan, Ritu Chaba
Sriraman Padmanabhan, Srivatsan Srinivasan, Wasim U. Quddus
Zhao Zhe and Adnan Aziz
Department of Electrical and Computer Engineering
University of Texas, Austin

Contact Person

Malay K. Ganai
University of Texas
Dept. of Electrical and Computer Engineering
ENS 530
Austin, TX 78712
Phone/Fax: 512-232-1883

E-mail: malay@godel.ece.utexas.edu

Abstract

Research in applied formal verification has been held back due to the lack of a meaningful set of examples. In this paper we describe a set of hardware designs that have been the subject of formal verification experiments using the VIS system. The examples are digital, synchronous, and control dominated. They cover a wide spectrum of the computational complexity essential to carry out verification experiments.

1 Introduction

Today the formal verification community suffers from a lack of meaningful and widely distributed examples for evaluating the performance of verification tools. Existing examples in the area of theorem proving are not finite state, and therefore, not suitable for this purpose. The examples in the VIS distribution are either toyish *(Eight Queens, Slider)* or trivial *(Dining Philosophers)*. More realistic hardware examples *(VIPER, 8085)* have little documentation and few property specifications [1].

The benefits of a set of examples are many. It will motivate the development of new algorithms. It will also facilitate comparisons across tools and provide case studies of verification methodologies for users.

In this work, we target control dominated hardware which has been found difficult to verify. We use $Verilog$ HDL for modeling hardware and a free version of a commercial $Verilog$ simulator called VeriWell [2]. We use the model checker provided with the VIS system for verification. The intermediate format read by VIS is $blif\_mv$ which can easily be converted to the SMV language [3] and other languages like S/R, the input language for $COSPAN$ [4].

We wrote the examples presented in this paper as term projects for a formal verification class offered by the Department of Electrical and Computer Engineering at the University of Texas, Austin [5]. The focus of the class was on symbolic model checking. The examples were collected by posting requests on the newsgroups *comp.lsi.cad* and *comp.arch* and mailing to VIS users and the CAV list. Many people in academia and industry showed an interest in these examples [6]. We implemented eight examples in groups of one, two or three persons.

2 Examples

We now describe the example suite.

2.1 MSI Cache Coherence Protocol

This is a model of an MSI (Modified Shared Invalid) cache coherence protocol to keep data consistent in a multiprocessor system with a cache coherent shared-memory architecture. The protocol is a bus based scheme in which individual processors *snoop* on all other processors' activities over a shared bus and respond accordingly. The response of each processor is determined by a set of well defined conditions implemented as a 3-state finite state machine [7].

2.2 PCI Local Bus

The PCI Local Bus is a high performance 32-bit or 64-bit bus with multiplexed address and data lines. It is used as an interconnect mechanism between peripheral controller components or add-in boards and processor/memory systems. This example contains a scaled down model with a 3-bit wide Data Bus, one Master and one Target module. Memory Read/Write and IO Read/Write transactions have been implemented [8].

2.3 PI-Bus Protocol

The PI-Bus is a high speed industry standard on-chip bus for use in micro-controllers and systems on a chip. The bus interface is between the bus on one side and the PI-Master agent on the other; a bus operation is an exchange of one or more multiple data between a master and the addressed slave. This example has two variants. One is the single master-single slave mode and the other is the multiple master mode [9].

2.4 MESI Cache Coherence Protocol

The Pentium Pro is tuned for a multi-processor environment with pipelining. This example models a scaled down version of the split-transaction bus and implements the MESI (Modified Exclusive Shared Invalid) Cache Coherence Protocol for 2 SMP processors [10].

2.5 MPEG System Decoder

The MPEG system decoder accepts an MPEG bit stream as input and demultiplexes it after extracting timing information. Data streams so produced are fed into audio and video decoders. This example implements this decoder as a series of modules each of which parses a specific part of the input bit stream [11].

2.6 Instruction Fetch Control Module of the Torch Microprocessor

TORCH is an experimental superscalar architecture developed at Stanford University. This example implements the Instruction Fetch Module. However, only the Instruction Fetch Control Module was verified, with the remaining modules, Instruction Fetch Datapath and Program Counter Unit Datapath, modeling the environment; this module accomplishes important signalling for the overall system [12].

2.7 DLX

This example implements the simple non-pipelined DLX processor (SDLX) and the pipelined DLX processor (PDLX) with a hazard detection unit. The design was scaled down for verification [13].

2.8 Blackjack

This example implements the game of Blackjack. It simulates both the dealer's and the player's actions with two different player strategies [14].

2.9 PowerPC 60x Bus Interface

This example contains behavorial model of the PowerPC Bus Interface derived from [15]. The 60x bus is designed to allow multiple processors to share the same memory map.

3 Results

The design size and the verification results for each example are tabulated (see Table 1). The examples PCI, MSI and DLX have been scaled down (without parameters). However, the scaling up of these examples should not be very difficult. For the Blackjack example we could not come up with any CTL property; coming up with suitable formulas is a challenge.

Table 1: Design Size and Verification Results

Example	*Variants*	*Gates*	*Latches*	*Properties*	*Result*
MSI	2 Processors	795	35	liveness, sharing	verified on scaled-down
	3 Processors	1175	48		
PCI Bus	Normal bus termination	5542	295	arbitration, deadlock, livelock	verified on scaled-down
	Abnormal bus termination	5584	295		
PI Bus	One-Master One-Slave	4183	225	bus-request, bus-grant, bus-contention	verified not exhaustively
	Multi-Master One-Slave	6541	307		
MESI	Fixed Opcodes	5721	298	liveness, cache coherency, transaction order	verified
	Var Opcodes for one				partially verified
	Var Opcodes for both				not verified
MPEG	-	10,983	559	deadlock, reseta-bility, state transition	verified separate modules; composite design not verified
Ifetch	-	1023	59	state transitions, signalling	several invariants were verified
DLX	SDLX	2725	311	liveness, deadlock	verified control unit
	PDLX	4292	559		not verified
Black-jack	Single player	4919	173	no significant property found	only simulated
	Double player	9225	248		
PPC 60x Bus	2 CPU	2843	47	liveness, arbitration, complex timing	simulated, partially verified
	1 CPU	1546	28		

4 Experiences

We observed that simulation is a useful method for finding low level bugs before beginning formal verification. The subset of *Verilog* supported by *vl2mv* is very restricted; this posed hurdles during modeling. Moreover, often code compiled by *vl2mv* is not accepted by VIS. The simulator in VIS is not user friendly; we found VeriWell quite helpful for simulation [2]. We faced problems as VIS does not support multiple clocks. Hence, some designs had to use gated clocks. However, we did find that VIS could be used as an excellent debugging utility. By verifying trivial formulas on VIS one could trace the logic flow in the design and catch errors.

5 Availability of Examples

The above examples are available at the following site as a single tar file:

http://godel.ece.utexas.edu/texas97-benchmarks/

Each example is provided with a document giving the specifications, block diagrams, properties, scalability, verification methods, results and inferences. Files giving *Verilog* and *blif_mv* (encoded and symbolic) descriptions, and CTL formulas are also provided.

6 Future Work

We found that some designs could not be handled by VIS because of the state space explosion. We believe that these pose a challenge for the tool developer. The examples which were scaled down due to the state explosion problem, can easily be scaled up as required. To compare the performance of different tools, a common intermediate language format is required. The development of such a format, or translators between existing formats is another problem that must be addressed.

7 Acknowledgements

We would like to thank Tom Shiple and Carl Pixley for their helpful and constructive comments.

References

[1] VIS Home page, `http://www-cad.eecs.berkeley.edu/~vis`.

[2] Wellspring Solutions Home Page, `http://www.wellspring.com/download.htm`.

[3] K. L. McMillan,"Symbolic Model Checking: An Approach to the State Explosion Problem," CMU Tech Rpt. CMU-CS-92-131.

[4] Robert P. Kurshan,"Analysis of Discrete Event Coordination," *Proceedings of the REX Workshop on Stepwise Refinement of Distributed Systems, Models, Formalisms, Correctness,* vol. 430, May 1989.

[5] EE 382m - Formal Verification, `http://www.ece.utexas.edu/~adnan/course/verif-97/index.html`.

[6] Projects for EE382m, `http://www.ece.utexas.edu/~adnan/course/verif-97/projects/index.html`.

[7] B. S. Mitchell,"Managing Cache Coherence in Multiprocessor Computer Systems," `http://www.mcs.drexel.edu/~bmitchel/research/CacheCoherence/CacheCoherence.html`.

[8] S. Campos, E. Clarke, W. Marrero, and M. Minea,"Verifying the Performance of the PCI Local Bus using Symbolic Techniques," *ICCAD,* 1995.

[9] Open Microprocessor Systems Initiative, `http://www.omimo.be/public/data/_omi324.htm`.

[10] T. Shanley, *Pentium Pro Processor System Architecture* MindShare, Inc.: Addison Wesley, April 1997.

[11] J. L. Mitchel, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall, *MPEG Video Compression Standard,* ITP Publications, 1997.

[12] Stanford's TORCH Project, `http://www-flash.stanford.edu:80/torch/`.

[13] High Performance Computer Architecture, `http://www.icaen.uiowa.edu/~hpca/`.

[14] Blackjack, `http://jvm.com/thop/`.

[15] M. Kaufmann and C. Pixley, "Interwined Development and Formal Verification of a 60x Bus Model," *ICCD,* Oct 1997.