Accelerating Boolean Implications with FPGAs

Kolja Sulimma, Wolfgang Kunz

March 11, 1999

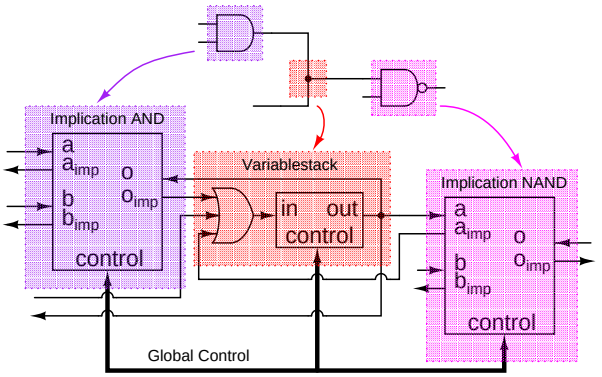

Figure 1: Gates to nodes

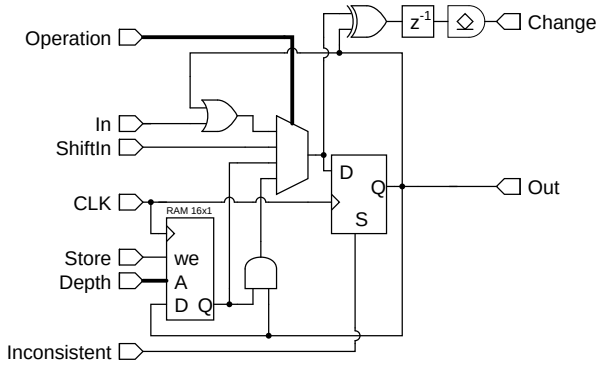

Figure 2: Processing element for a boolean variable

Abstract

The article presents the FPGA implementation of an algorithm that computes all indirect implications in a boolean network. The research was performed as a master thesis at University of Frankfurt, [4].

Computing all implications for a partial value assignment in a boolean network has many applications, mainly in the area of synthesis and verification of digital circuits. It can, for example, be used to accelerate the generation of test vectors and to improve the optimization of digital circuits. Satisfiability problems can be solved by computing all implications of an value assignment at the primary outputs. [3] presents an algorithm that computes all implication by performing an AND/OR enumeration on the gates with unjustified value assignments.

To take full advantage of the possible fine grain parallism of FPGAs, a circuit generator was writtien using the BOOM Package of UC Berkeley [1] that transforms a boolean network into a network of small processing elements. One element for each gate and each variable respectivly as shown in figure 1. Each processing elemt contains the logic necessarry to perform the operations requiered by the algorithm and a stack that allows to save the state of the element up to an recursion depth of 16.

The recursiv algorithm is rather complex for a harware realisation and therefore the implementation is an interesting example for the potential of reconfigurable computing beyond simple systolic algorithms.

Figure 2 shows the schematic of a processing element for a boolean variable. Each processing element requires four XC4K logic blocks. This means that with todays FPGAs implications can be performed in circuits with up to 1000 Gates in a single Chip. Arrays of multiple FPGAs can be used with only a minor performance impact, as the implementation still works correctly if the delay and bandwidth

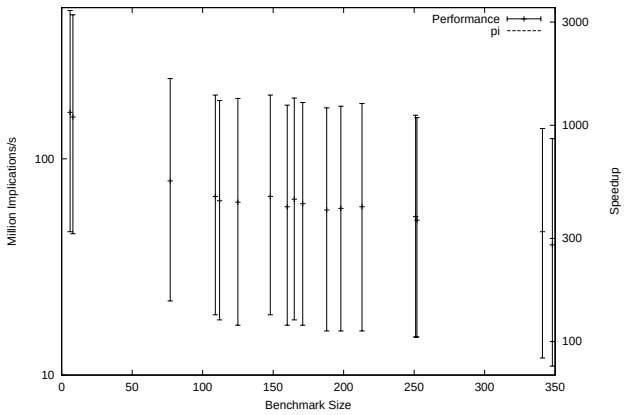

Figure 3: Performance of the accelerator by benchmark size

vary from connection to connection.

The FPGA configuration depends on the input network and therefore must be synthesized on demand for each boolean network that is to be examined.The circuit can be used without changes for multiple initial value assignments.

The performance of this approach has been evaluated for ISCAS benchmarks with up to 600 gates. For each benchmark a configuration for XC4000XV FPGAs was generated using the BOOM generator and the Xilinx Foundation synthesis tools. The performance of the various operations required by the algorithm was than analyzed using the Foundation static timing analyzer.

Compared to a 220 MHz Ultrasparc workstation a speedup of two orders of magnitudes was achieved as shown in figure 3. The Diagram shows quite large error bars because the parallism of the computation and with it the achievable speedup depends highly on the initial value assignment. The center values are typical values for satisfiability problems. The lower values are valid for worst case assumptions that were never observed during our experiments. The upper bound are results for assignments close to high fanout nodes.

These results show, that the computation of indirect implications can been sped up several hundred times by reconfigurable hardware if the synthesis times are neglected. With todays tools however this approach is not practical because the synthesis times for the larger examples alone exceeded the typical running time of the software version of the algorithm.

Therefore the development of very fast synthesis algorithms and reconfigurable architechtures that support fast synthesis are essential fore reconfigurable computing. In the above case a several times lower performance can easily be accepted if the synthesis takes only a couple of minutes. MITs TSFPGA [2] and UC Berkeleys BRASS project have some promissing results in that area.

References

[1] Michael Chu, Kolja Sulimma, Nick Weaver, Andre DeHon and John Wawrzynek: "Object Oriented Circuit-Generators in Java"; *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines, 1998*
`http://www.cs.berkeley.edu/projects/brass/documents/Generators_FCCM98.html`

[2] Andre DeHon: "Reconfigurable Architechtures for General-Purpose Computing"; A.I. Technical Report No. 1586, Massachusets Institute of Technology, Artificial Intelligence Laboratory, 1996.
`http://www.ai.mit.edu/people/andre/phd.html`

[3] Wolfgang Kunz and Dominik Stoffel: "Reasoning in Boolean Networks"; Kluwer Academic Publishers, 1997. ISBN 0-7923-9921-8

[4] Kolja Sulimma: "Berechnung von Implikationen in Booleschen Netzen mit FPGAs", Universität Frankfurt, 1999. ISBN 3-933966-00-0
`http://www.prowokulta.org/verlag/`

[5] Peixing Zhong, Margaret Martonosi, Pranav Ashar and Sharad Malik: "Accelerating Boolean Satisfiability with Configurable Hardware"; *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines*, 1998
`http://www.ee.princeton.edu/~mrm/pubs.html`