

A General Computational Framework and Simulations of Branching Programs of Boolean Circuits Using Higher Order Logic (HOL) Software - An Insight into ECAD Tool Design Paradigm

D. N. T. Kumar^{1,2,3,4,5,6,7} & Qufu Wei^{1,2}

¹ Laboratory of Nano-Technology/Sensor Technology R&D Program, Jiangnan University, Wuxi, P. R. China

² Key Laboratory of Eco-Textiles, Graduate School of Textiles & Clothing, Jiangnan University, Wuxi, P. R. China

³ Ministry of Education, Jiangnan University, Wuxi, 214122, Jiangsu Province, P. R. China

⁴ Department of Chemical Engg, BGU, Israel

⁵ Department of Physics, BGU, Israel

⁶ The Ilse Katz, Center for Meso and Nanoscale Science and Technology, BGU, Israel

⁷ Ben-Gurion University of the Negev, Beersheva, Israel

Correspondence: D. N. T. Kumar, Jiangnan University, Wuxi, 214122, China and Ben-Gurion University of the Negev, BeerSheva, 84105, Israel. E-mail: tejdkn@gmail.com; qfwei@jiangnan.edu.cn

Received: August 8, 2012 Accepted: September 4, 2012 Online Published: September 13, 2012

doi:10.5539/cis.v5n6p6

URL: <http://dx.doi.org/10.5539/cis.v5n6p6>

Abstract

Integrated optics and Optical computing are now a mature technology offering many types of devices and manufacturing techniques. Recent breakthroughs in the field of silicon photonics showed low-loss insulators, passive wave guide devices, high speed optical switches, detectors, silicon lasers, and silicon amplifiers, optical amplifiers etc. These devices have provided the possibility to construct CMOS compatible optical circuits with low power consumption, high bandwidth and low latencies. A critical component that we intend to focus is on considering Branching Programs as CAD tools for the design of future electronics. Hence, BPs as viable means to accomplish the task of propelling the R&D of Electronics, are considered and simulated using HOL software based on boolean theory. We believe our research is one of the remarkable pioneering efforts, into the promising aspects of Branching Programs as CAD Tools.

Keywords: branching programs (BP), Boolean circuits, integrated optics, lemmas, HOL, CAD tools, optical computing

1. Introduction and Inspiration

Before taking the readers to the introduction of BP based CAD tool design and analysis, we are interested in beginning with a brief introduction to the concepts involved in framing our paper.

The basis of motivation for this short communication/technical note/tutorial, is based on one of the promising papers presented by Anthony Fox, Computer Laboratory, University of Cambridge, UK

Titled: “*ARM Formalized in HOL*” (Anthony Fox, 2012). Hence in our current work, we developed a simple general computational framework and highlight HOL as the design, computational and verification tool of electronics research and development.

1.1 Higher Order Logic (HOL)

“HOL is one of a family of theorem provers that share a common design called the ‘LCF architecture’. The LCF architecture is so named because its first use was in a theorem prover called LCF. Users of LCF architecture tools prove theorems in the logic supported by the tool (the object language) by writing programs to construct a proof using a programming language (the meta-language). The object language of the HOL system is a classical higher-order logic based on Church’s simple theory of types. The meta-language, as with other LCF provers, is ML. The original motivation for building HOL, as a tool for assisting in the specification and verification of digital

hardware, the higher-order logic supported by HOL is particularly well suited to such problems. However, the logic is a general one and HOL has now been applied to many different kinds of problems” (Al-Ruwaihi & Hindy, 1997; Phoenix optical simulation software field designer (mode solver) and optodesigner (beam propagation methods); Spice simulations Wikipedia; Beamprop product overview; Alam & Lake, 2005; Bakoglu & Meindl, 1985; Balijepalli, Sinha, & Cao, 2007; Barrington, 1986; Ben-Asher & Meisler, 2006; Ben-Asher & Rotem, 2008; Ben-Asher & Shochat, 2008; Ben-Asher, Citron, & Haber, 2004; Ben-Asher, Peleg, & Schuster, 1992; Bertacco, Minato, Verplaetse, Benini, & Micheli, 1997; Cai & Lipton, 1989; Dong, Preble, & Lipson, 2007; Feitelson, 1988).

1.2 Branching Program/s (BP)

“A branching program on the variable “x” is a finite directed acyclic graph with one source node and sink nodes partitioned into two sets, Accept and Reject. Each non-sink node is labeled by a variable x_i and has two outgoing edges labeled 0 and 1 respectively. The length of the program is the maximum length of any such path. We are only going to consider layered branching programs of length. Here the nodes are partitioned into ` sets and edges only go from one layer to the next. The width of a layered branching program is the maximum number of vertices in any layer. A branching program can be thought of as a space-bounded model of computation where space= \log (width); from each state, we just look at 1 bit of the input. This is a clean model of space-bounded computation which abstracts from model-dependent Turing-machine issues such as keeping track of the position of the head on the input tape. One can have similar branching programs for the parity function. It can be shown that every function on n bits can be computed by a branching program of width 3 and exponential length” (Hunsperger, 2002; Hussein, Nounou, Saada, Atef, & Khalil, 2006; Jang, Park, & Prasanna, 1992; Taki, 2000; Lattner, 2002; Mehlhorn & Nher, 1995; Moreinis, Morgenshtein, Wagner, & Kolodny, 2006; Morgenshtein, Fish, & Wagner, 2002; Morgenshtein, Friedman, Ginosar, & Kolodny, 2008; Nakano & Wada, 1998; Nishihara, Haruna, & Suhara, 1987; Okayama, Okabe, Kamijoh, & Sakamoto, 1999; Reed & Knights, 2004; Shi, Wa, Miller, Pamulapati, & Cooke, 1995; Isabelle, 2012; Anthony Fox, 2012; HOL: The Higher Order Logic Theorem Prover; Barrington’s Theorem, 2009).

2. Deduction of Branching Program/s for Boolean Circuit/s

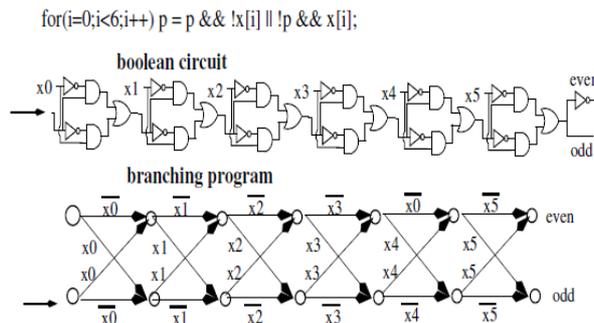


Figure 1. Branching program for a Boolean circuit-even/odd parity function

Explanation of Figure 1: In Figure 1, we designed a simple boolean circuit and deduced its equivalent branching program, using the established mathematical and computational paradigms.

For more information, please refer to the references listed below.

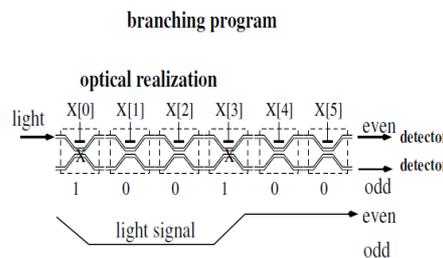


Figure 2. Suggestion of optical switches based implementation of even/odd parity circuit for information processing

Explanation of Figure 2: In Figure 2, a simple even/odd parity circuit for driving or processing the information is shown. We suggested the even/odd parity circuit using optical switches which hold a great promise for the

futuristic tendencies in electronics and optical computing systems. In here we consider implementations of BPs, that are based on optical switches fabricated using Integrated Optics techniques. Optical switching devices (Feitelson, 1988) is an advance research area including many types of devices that can switch light between different wave-guides or directly redirect light beams in free space. Many works describe optical switching networks (prototypes and devices), e.g., (Okayama, Okabe, Kamijoh, & Sakamoto, 1999) describing a 32X32 optical banyan switching network.

3. Approximate Simulation of Branching Programs of Boolean Circuits Using HOL Software

In this section we briefly present our idea of BP implementation using HOL Software, first in the form of an approximate flowchart diagram and then present a simple HOL notation, to solve boolean functions using “Isabelle HOL Software system” (beam propagation methods); Spice simulations Wikipedia; Beamprop product overview; Alam & Lake, 2005; Bakoglu & Meindl, 1985; Balijepalli, Sinha, & Cao, 2007; Barrington, 1986; Ben-Asher & Meisler, 2006; Ben-Asher & Rotem, 2008; Ben-Asher & Shochat, 2008; Ben-Asher, Citron, & Haber, 2004; Ben-Asher, Peleg, & Schuster, 1992; Bertacco, Minato, Verplaetse, Benini, & Micheli, 1997; Cai & Lipton, 1989; Dong, Preble, & Lipson, 2007; Feitelson, 1988; Ganguly, Samanta, Das, & Biswas, 2002).

3.1 Flow Chart Analysis of the Main Idea in Our Paper

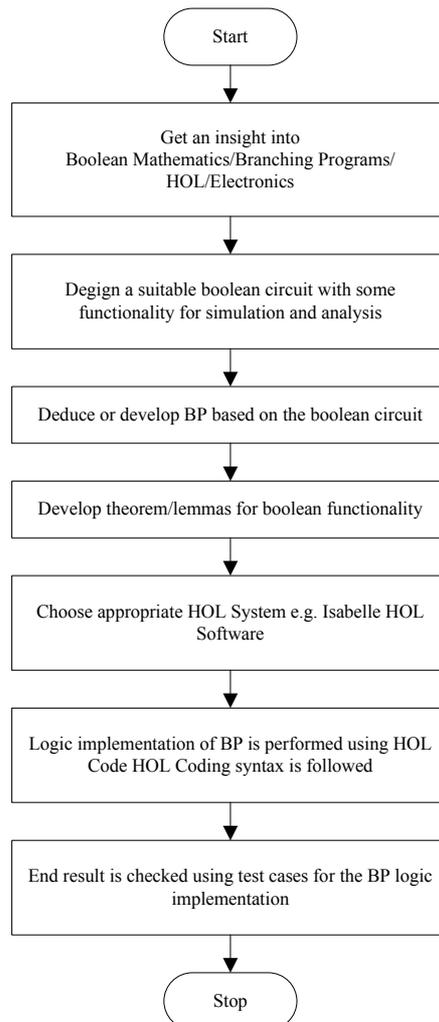


Figure 3. Flow chart summary of the entire idea presented in our paper

3.2 Simulation Code Using HOL Specification

(* Title: Bool.thy/Implemented using Isabelle HOL System

Author: Nirmal, 2012, Jiangnan University, China/Ben-Gurion University, Israel

The HOL Code mentioned below, is just to provide some guidelines for the readers. Precise implementation is

possible for the above mentioned Boolean circuit or Optical circuit by defining Lemmas/Theorems and simulating them using HOL system (Isabelle, 2012) *).

header{*Booleans in Branching Programs*}

theory Bool imports pair begin

abbreviation

even ("1") where

"1 == succ(0)"

abbreviation

odd ("2") where

"2 == succ(1)"

*Text {*2 is equal to bool, but is used as a number rather than a type.*}*

definition "bool == {0,1}"

definition "cond(b,c,d) == if(b=1,c,d)"

definition "not(b) == cond(b,0,1)"

definition

"and" :: "[i,i]=>i" (boolean condition) where

"a and b == cond(a,b,0)"

definition

or :: "[i,i]=>i" (boolean condition) where

"a or b == cond(a,1,b)"

definition

xor :: "[i,i]=>i" (boolean condition) where

"a xor b == cond(a,not(b),b)"

lemmas bool_defs = bool_def cond_def

lemma singleton_0: "{0} = 1"

by (simp add: succ_def)

(* Introduction Rules *)

lemma bool_1I [simp,TC]: "1 \<in> bool"

by (simp add: bool_defs)

lemma bool_0I [simp,TC]: "0 \<in> bool"

by (simp add: bool_defs)

lemma one_not_0: "1 \<noteq> 0"

by (simp add: bool_defs)

(** 1=0 ==> R **)

lemmas one_neq_0 = one_not_0 [THEN not E]

lemma boolE:

"[| c: bool; c=1 ==> P; c=0 ==> P |] ==> P"

by (simp add: bool_defs, blast)

(** cond **)

(*1 means odd*)

Note to the Readers: Since there is plenty of literature available, on the fundamentals of Boolean algebra/Theorems/Lemmas/HOL/Branching Programs, we are not discussing these issues in detail here. Readers are advised to study the references mentioned in this paper for further information, thanks for your understanding.

4. Discussion

Thus we could successfully analyze and perform research into Branching Programs as CAD tools for future electronics and nanotechnology. The utility of Branching programs is quite useful in the synthesis of both analog circuits and optical circuits. Though we highlighted an “Optical Circuit” for information processing and computation of even/odd parity circuit, we are not discussing in detail as such advanced topics like Photonics and Optical Computing, which deserve another research paper on an in-depth basis. Hence we are of the belief that our current research work is one of the pioneering efforts in this promising domain of Branching Programs which is interdisciplinary in nature (Mehlhorn & Nher, 1995; Moreinis, Morgenshtein, Wagner, & Kolodny, 2006; Morgenshtein, Fish, & Wagner, 2002; Morgenshtein, Friedman, Ginosar, & Kolodny, 2008; Nakano & Wada, 1998; Nishihara, Haruna, & Suhara, 1987; Okayama, Okabe, Kamijoh, & Sakamoto, 1999; Reed & Knights, 2004; Shi, Wa, Miller, Pamulapati, & Cooke, 1995; Isabelle, 2012; Anthony Fox, 2012; HOL: The Higher Order Logic Theorem Prover; Barrington’s Theorem, 2009).

5. Conclusion and Future Perspectives

The potential impact of the above discussed issues concerning BPs to synthesis future electronics and nano-devices, show that integrated optical synthesis of BP can be automated to a similar level as that of VLSI circuits. Reveal new optimization problems involved with optimizing integrated optic synthesis of BP. Promote and expose the advantages of optical BP synthesis compared to CMOS FET circuits. This research if successful can be possibly extended to an implementation of a light based CPU, adding a new direction to the quest of techniques for building pure light based computers. It is a new direction for high level synthesis systems targeting BPs synthesis instead of Verilog. The ability to speedup these devices using large BPs circuits implemented using integrated optics, can have a profound impact on digital computer systems and their designs with clear economical implications.

Information on HOL Software Used

“*Isabelle* is a generic proof assistant. It allows mathematical formulas to be expressed in a formal language and provides tools for proving those formulas in a logical calculus. Isabelle is developed at University of Cambridge (Larry Paulson), Technische Universität München (Tobias Nipkow) and Université Paris-Sud (Makarius Wenzel). See the Isabelle overview for a brief introduction” (Isabelle, 2012).

Acknowledgements

We, sincerely thank all the members, involved in making this paper a possibility. Further, we thank Jiangnan University, China and Ben-Gurion University of the Negev, Israel, for their research support and for providing conducive research environment. We finally thank the authors, of some of the published papers, who willingly gave us permissions to reproduce some materials, from their research papers and also encouraged us to write this paper. We declare further that, we have no competing financial interests in the method presented in this paper. The authors strictly abide by Open Source Software policies and agreements.

References

- Alam, K., & Lake, R. K. (2005). Leakage and performance of zero-Schottky-barrier carbon nanotube transistors. *Journal of Applied Physics*, 98(6), 4307-4314. <http://dx.doi.org/10.1063/1.2060962>
- Al-Ruwaihi, K. M., & Hindy, M. A. (1997). Analysis of a digital microstrip optical switch: a novel method. *Appl. Opt.*, 36, 1213-1217. <http://dx.doi.org/10.1364/AO.36.001213>
- Anthony Fox. (2012). Retrieved from <http://www.cl.cam.ac.uk/~acjf3/>
- Bakoglu, H. B., & Meindl, J. D. (1985). Optimal interconnection circuits for vlsi. *IEEE Trans. On Electron Devices*, ED-32, 903-909. <http://dx.doi.org/10.1109/T-ED.1985.22046>
- Balijepalli, A., Sinha, S., & Cao, Y. (2007). Compact modeling of carbon nanotube tran-sistor for early stage process-design exploration. In ISLPED '07: Proceedings of the 2007
- Barrington, D. A. (1986). Bounded-width polynomial-size branching programs recognize exactly those languages in ncl. In STOC '86: Proceedings of the eighteenth annual ACM symposium on Theory of computing, 1986.
- Beamprop product overview. Retrieved from <http://www.rsoftdesign.com/products.php>
- Ben-Asher, Y., Citron, D., & Haber, G. (2004). *Overlapping memory operations with circuit evaluation in reconfigurable computing*. In 18th International Parallel and Distributed Processing Symposium (IPDPS'04) - Workshop 3.
- Ben-Asher, Y., & Meisler, D. (2006). *Towards a source level compiler: Source level modulo scheduling*. In 5th

- Workshop on Compile and Runtime Techniques for Parallel Computing (CRTPC) Ohio.
- Ben-Asher, Y., Peleg, D., & Schuster, A. (1992). *The complexity of reconfiguring network models*. In Israel Symposium on Theory of Computing Systems, pp. 79-90.
- Ben-Asher, Y., & Rotem, N. (2008). *System racer: Synthesis for variable pipelined multiplications and other operations*. In International Symposium on System-on-Chip 2008 Tampere, Finland.
- Ben-Asher, Y., & Shochat, E. (2008). *Finding the best compromise in compiling compound loops to verilo*. In IEEE Computer Society Annual Symposium on VLSI.
- Bertacco, V., Minato, S., Verplaetse, P., Benini, L., & Micheli, G. D. (1997). *Decision diagrams and pass transistor logic synthesis*. Technical report.
- Cai, J. Y., & Lipton, R. J. (1989). Subquadratic simulations of circuits by branching programs.
- Dong, P., Preble, S. F., & Lipson, M. (2007). All-optical compact silicon comb switch. *Optics Express*, 15(15), 9600-9605. <http://dx.doi.org/10.1364/OE.15.009600>
- Feitelson, D. G. (1988). *Optical computing*. MIT press.
- Ganguly, P., Samanta, B., Das, S., & Biswas, J. C. (2002). Design, fabrication and preliminary characterisation of zklinbo, directional coupler switch. *Defence Science Journal*, 52(2), 201-203.
- Himsolt, M. (1994). *Graphed: A graphical platform for the implementation of graph algorithms*. In Graph Drawing (GD '94).
- HOL: The Higher Order Logic Theorem Prover. (2012). <http://cs.anu.edu.au/student/comp8033/hol.html> [37]
- Barrington's Theorem. (2009). Retrieved from <http://www.ccs.neu.edu/home/viola/classes/gems-08/lectures/le11.pdf>
- Hunsperger, R. G. (2002). *Integrated Optics*. Springer-Verlag.
- Hussein, A., Nounou, A., Saada, N., Atef, D., & Khalil, D. (2006). *Spice modeling of free-space optical systems*. In IEEE International Behavioral Modeling and Simulation Workshop. <http://dx.doi.org/10.1109/BMAS.2006.283475>
- Isabelle. (2012). Retrieved from <http://www.cl.cam.ac.uk/research/hvg/isabelle/>
- Jang, J., Park, H., & Prasanna, V. K. (1992). *An optimal multiplication algorithm on reconfigurable mesh*. In Proc. Symp. on Parallel and Distributed Processing, pp. 381-391.
- Lattner, C. (2002). *LLVM: An Infrastructure for Multi-Stage Optimization*. Master's thesis, Computer Science Dept., University of Illinois at Urbana-Champaign, Urbana, IL.
- Mehlhorn, K., & Nher, S. (1995). Leda a platform for combinatorial and geometric computing. *Communications of the ACM*, 38, 96-102. <http://dx.doi.org/10.1145/204865.204889>
- Moreinis, M., Morgenshtein, A., Wagner, I., & Kolodny, A. (2006). Logic gates as repeaters (lgr) for area-efficient timing optimization. *IEEE Trans. on Very Large Scale Integration Systems*, 14(11), 1276-1281. <http://dx.doi.org/10.1109/TVLSI.2006.886400>
- Morgenshtein, A., Fish, A., & Wagner, I. (2002). Gate-diffusion input (gdi): a power-efficient method for digital combinatorial circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 10, 566-581.
- Morgenshtein, A., Friedman, E. G., Ginosar, R., & Kolodny, A. (2008). *Timing optimization in logic with interconnect*. In The Intl Workshop on System Level Interconnect Prediction (SLIP). <http://dx.doi.org/10.1145/1353610.1353615>
- Nakano, K., & Wada, K. (1998). Integer summing algorithms on reconfigurable meshes. *Theoretical Computer Science*, 197, 57-77. [http://dx.doi.org/10.1016/S0304-3975\(97\)00007-8](http://dx.doi.org/10.1016/S0304-3975(97)00007-8)
- Nishihara, H., Haruna, M., & Suhara, T. (1987). *Optical Integrated Circuits*. McGraw-Hill Optical and Electro-Optical Engineering Series.
- Okayama, H., Okabe, Y., Kamijoh, T., & Sakamoto, N. (1999). Optical switch array using banyan network. *IEICE Transactions on Communications*, E82-B(2), 365-372.
- Phoenix optical simulation software fielddesigner (mode solver) and optodesigner (beam propagation methods). Retrieved from <http://www.rsoftdesign.com/products.php?sub=component+design&itm=beamprop>
- Reed, G. T., & Knights, A. P. (2004). *Silicon Photonics*. John Wiley & Sons, Inc.

<http://dx.doi.org/10.1002/0470014180>

Shi, S., Wa, P. L. K., Miller, A., Pamulapati, J., & Cooke, P. (1995). Multi quantumwell zerogap directional coupler with disordered branching waveguides. *Applied Physics Letters*, 66(1), 79-81.
<http://dx.doi.org/10.1063/1.114151>

Spice simulations Wikipedia. Retrieved from <http://en.wikipedia.org/wiki/spice>

Taki, K. (2000). *A survey for pass-transistor logic technologies*. In ASPDAC-98.