Zhiyang Ong and Andrew Mattheisen
zhiyango@usc.edu & amattheisen@gmail.com

Erratum

Table of Contents

Adder/Subtractor Design

The Synopsys DesignWare intellectual property (IP) library consists of high performance IP blocks for system development and integration to reduce development time, and time-to-market (Cohen et al, 1996). Given the short design time of this processor project and the high performance requirement, we decided to utilize the virtual microarchitecture library from DesignWare IP to implement the adder design (Synopsys, 2001; Synopsys, 2007).

This would save us time from implementing advanced adder designs, using Verilog, in structural RTL. It would also save us numerous man-hours from iterating the design process to verify that the design functions correctly, and optimize the design for high performance within our aggressive schedule. Hence, the addition operator "+" in Verilog was used for addition, and the Synopsys Design Compiler would interact utilize adder components from the DesignWare IP library to build the high performance adders.

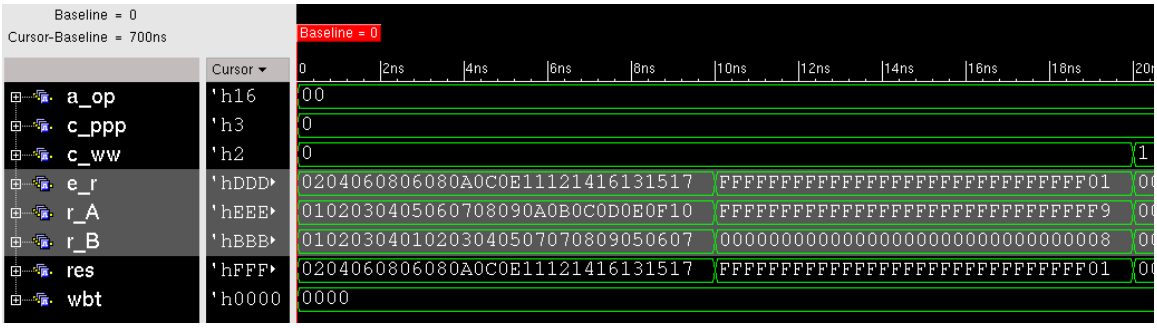
If we had more time to design the Troy WideWord Processor, we would consider the following adder designs and design techniques. For example, Carry-Save-Adders (CSAs) can be optimized for performance across multiplexors, design boundaries, and multiplications. While Synopsys Design Compiler used to implement some of the steps involved, a significant amount of work would be needed to implement the optimization techniques (Kim et al, 1998a; Kim et al, 1998b; Kim and Um, 2000a; Kim and Um, 2000b).

In addition, soft cores and testing structures for adders, and other types of circuitry performing arithmetic functions, can be automatically generated with electronic design automation (EDA) tools that contain their area, delay, and power characteristics. With such cores, we can select some of them for usage, or optimize some adder designs. If optimization of such adder designs cannot meet our performance goals, we can use other techniques to customize our adder design (Bakalis et al, 2006). This also applies to the development of other arithmetic circuitry.

If we pursued custom or semi-custom design of this processor with a greater amount of design time, we can consider carry-look-ahead adders (Koren, 1993), tree and prefix adders such as carry-select adders and carry-save adders (Ercegovac and Lang, 2004), Manchester Carry-Chain Adder, Square-Root Carry-Select Adder, and Logarithmic Lookahead Adder (Rabaey et al, 2003), Brent-Kung, Sklansky, Kogge-Stone, Han-Carlson, Knowles, and Ladner-Fischer tree adders (Weste and Harris, 2005). In addition, performance optimization of such adder designs can be carried out with pipelining, gate-sizing techniques (including the method of logical effort), and skew-tolerant domino logic (Weste and Harris, 2005).

Simulators for various adder designs mentioned above, along with other arithmetic circuits, can be obtained at http://www.ecs.umass.edu/ece/koren/arith/simulator/.

In our adder design, we had not used low-power design techniques to reduce power consumption, since it is not part of our project's objectives. However, if we had more design time, we would consider the use of power-aware RTL macro selection strategies to reduce power consumption in our adder design (Simone et al, 2006). Subsequently, trade-offs in area, delay, and power can be considered as we attempt to design fast adders to help us meet our project objectives.



Pre-synthesis simulation of some addition test cases

Pre-synthesis simulation of some addition test cases

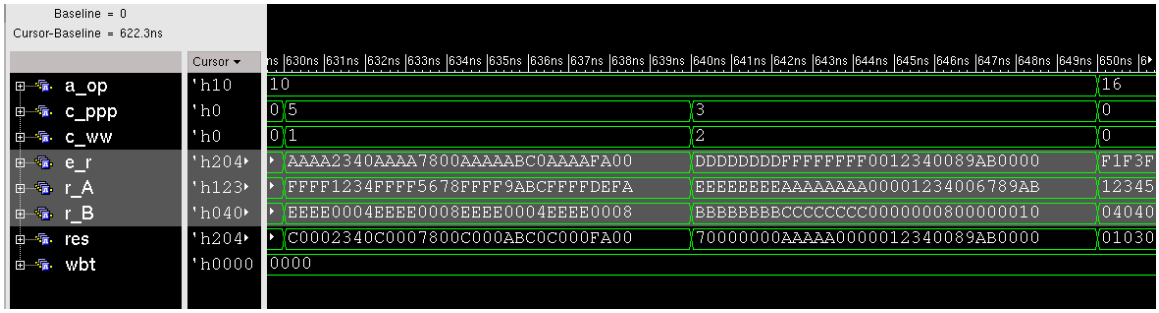
Shifter Design

Similar to the adder, the shifter was also implemented using the logical shift operators from Verilog. During synthesis of this shiter design, Design Compiler would utilize the virtual microarchitecture library from DesignWare IP to implement the shifters

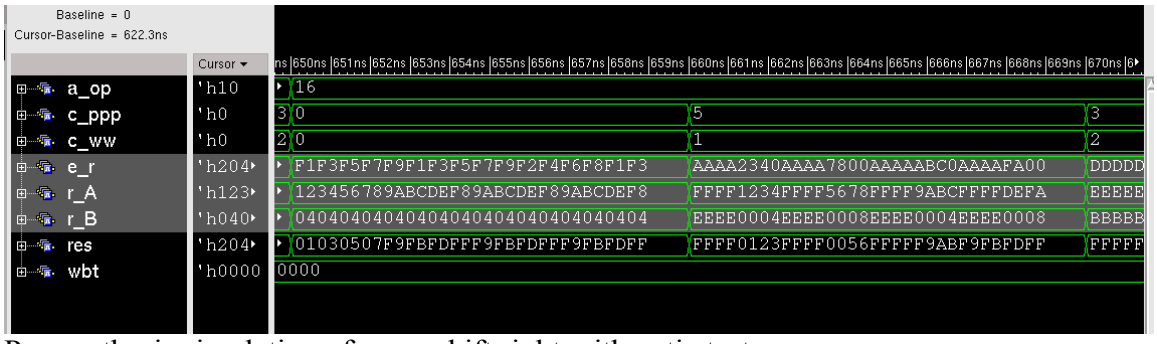
(Synopsys, 2001).

Likewise, with greater design time, we can implement structural RTL models of barrel shifter and logarithmic shifter (Rabaey et al, 2003), and array funnel shifter and multilevel funnel shifter (Weste and Harris, 2005).

From the synthesized report of the shifter, the total cell area is 437944, and its delay is 9.77 ns. Also, its total dynamic power is 434.7297 mW, and its cell leakage power is 711.5905 nW.



Pre-synthesis simulation of some shift left logical test cases



Pre-synthesis simulation of some shift right arithmetic test cases

Multiplier Design

As part of the project requirements, multipliers obtained from the synthesis of the Verilog multiplication operator, the asterisk "*", cannot be used. Hence, we had to implement a multiplier deisgn. However, for purposes of performance comparison, multipliers implemented using the asterisk operator were developed, and synthesized to compare their power, area, and delay characteristics with our multiplier design.

A basic sequential algorithm for multiplication is the shift-and-add multiplication algorithm (Koren, 1993; Ma and Taylor, 1990; Weste and Harris, 2005). This was used to implement the unsigned even and odd multiplication instructions. For the signed even and odd multiplication instructions, the Booth-Wooley multiplier was chosen instead (Weste and Harris, 2005). This is because algorithmic descriptions of the multiplier can be obtained, and implemented in behavioral RTL so that the Design Compiler can be used to do the logic minimization.

On the other hand, other multipliers that may be faster were usually described in terms of their logic blocks. Consequently, only the structural RTL models of such multipliers can be developed, and it would reduce the solution space for optimization by the Design Compiler. This is because Design Compiler can only do logic minimization and technology mapping, without behavioral synthesis (Khatri and Shenoy, 2006).

Given more design time, parallel, array, and sequential, which involve pipelining, multipliers can be used to improve performance. Examples of such multipliers include ripple-carry based array multiplier, carry-save array multiplier, and Wallace tree multiplier, (Rabaey et al, 2003; Parhami, 2001).

Also, EDA tools can be developed or tweaked to optimize our adder. For example, the allocation of carry-save-adders used in multiplier designs can be optimized (Um and Kim, 2001). Or, the Three-Dimensional Method (TDM) can be used to determine the Partial Product Reduction Tree (PPRT) design that is needed for delay minimization of parallel multipliers (Stelling et al, 1998).

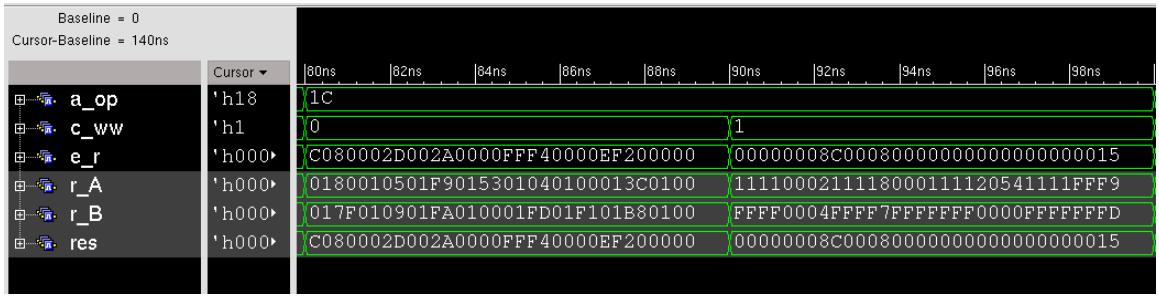
Table 1 Power, Area, and Delay Characteristics for Different Multiplier Designs

Name of Multiplier Design	Total Dynamic Power	Cell Leakage Power	Total Power Consumption	Total Cell Area	Data Arrival Time

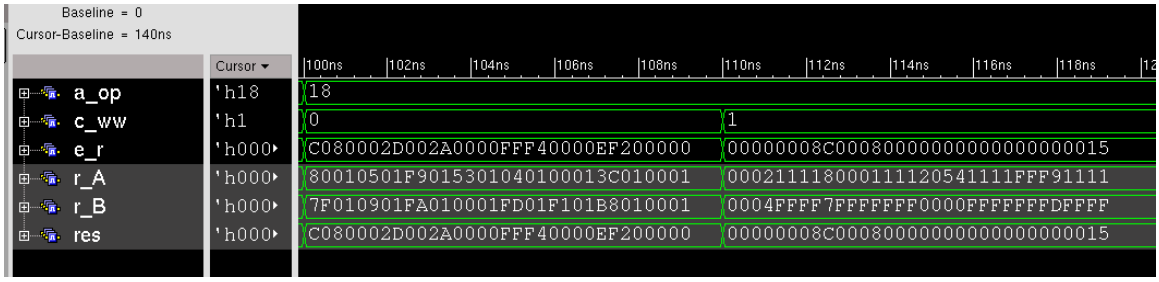
	(mW)	(mW)	(mW)		
Use of Verilog's asterisk, *, operator	426.1780	394.4543	820.6323	196835	11.06
Use of signed multiplier using Booth's					

Table 1 gives the power, area, and delay characteristics for different multiplier designs. The total cell area is based on the unit cell area in the technology library (Synopsys, 2002; Synopsys, 1999). For power consumption, two figures of merit are given IN milliwatts: total dynamic power and cell leakage power. In addition, for preliminary timing analysis and verification of its functionality, the data arrival time is the only statistic released publicly. This figure of merit is provided from the static timing analyzer of the Design Compiler, and is measured in nanoseconds (Synopsys, 1999).

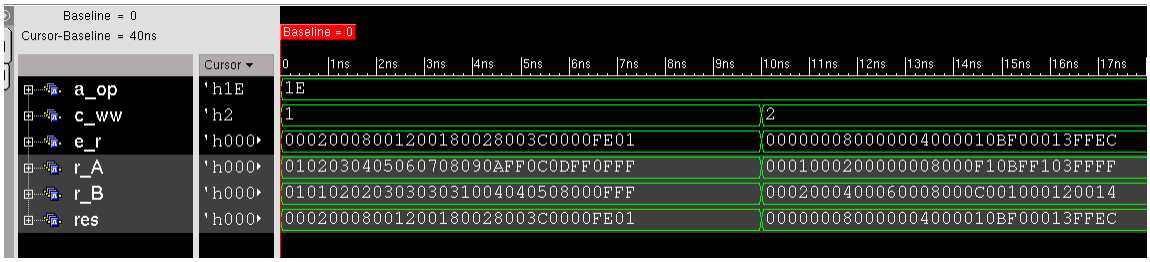
We note that the design compiler only estimates the dynamic (switching) and leakage power consumption. It ignores short-circuit and static power consumption, which form part of the total power consumption of the circuit/system (Kang and Leblebici, 2003). Thus, the total power consumption, which is the summation of dynamic and leakage power consumption, indicated in Table 1 does not include short-circuit and static power consumption.



Pre-synthesis simulation waveform for signed odd multiplication instructions.



Pre-synthesis simulation waveform for signed even multiplication instructions.



Pre-synthesis simulation waveform for unsigned odd multiplication instructions.

Reference

D. Bakalis, K. D. Adaos, D. Lymperopoulos, M. Bellos, H. T. Vergos, G. Ph. Alexiou, and D. Nikolos, “A Core Generator for Arithmetic Cores and Testing Structures with a Network Interface”, Journal of Systems Architecture, vol. 52, no. 1, January, 2006, pages 1-12.

Morris A. Cohen, Jehoshua Eliashberg, and Teck-Hua Ho, “New Product Development: The Performance and Time-to-Market Tradeoff”, Management Science, vol. 42, no. 2., February, 1996, pages 173-186.

Miloš D. Ercegovac and Tomás Lang, “Digital Arithmetic”, Morgan Kaufmann Publishers, San Francisco, CA, 2004, pages 85, 98.

Sung-Mo Kang and Yusuf Leblebici, “CMOS Digital Integrated Circuits: Analysis and Design”, 3rd Edition, McGraw-Hill, New York, NY, 2003, pages 482, 492.

Taewhan Kim, William Jao, and Steve Tjiang, “Arithmetic Optimization using Carry-Save-Adders”, Proceedings of the 35th Design Automation Conference, San Francisco, CA, June 15-19, 1998a, pages 433-438.

Taewhan Kim, William Jao, and Steve Tjiang, “Circuit Optimization Using Carry–Save–Adder Cells”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 17, no. 10, October, 1998b, pages 974-984.

Taewhan Kim and Junhyung Um, “A Timing-Driven Synthesis of Arithmetic Circuits using Carry-Save-Adders”, Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC 2000, Yokohama, Japan, Jan 25-28, 2000a, pages 313-316.

Taewhan Kim and Junhyung Um, “A Practical Approach to the Synthesis of Arithmetic

Circuits Using Carry-Save-Adders", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 19, no. 5, May 2000b, pages 615-624.

Israel Koren, "Computer Arithmetic Algorithms", Prentice Hall, Englewood Cliffs, NJ, 1993, pages 29-32.

Israel Koren [Online], "Computer Arithmetic Algorithms Simulator": The Algorithms; last viewed December 13, 2007, at <http://www.ecs.umass.edu/ece/koren/arith/simulator/>, 2005.

Gin-Kou Ma and Fred J. Taylor, "Multiplier policies for digital signal processing", IEEE ASSP Magazine, vol. 7, no. 1, January, 1990, pages 6-20.

Ann Maldonado-Vazquez, "Power – Performance Tradeoffs In Digital Arithmetic Circuits", Report for the Summer Undergraduate Program in Engineering Research at Berkeley (SUPERB), Summer, 2003, College of Engineering, University of California, Berkeley.

Behrooz Parhami [Online], Instructor's Manual for Computer Arithmetic: Algorithms and Hardware Designs: Volume 2: Presentation Material: Textbook on Computer Arithmetic: Presentations, 2001; last viewed on December 10, 2007, at

<http://www.ece.ucsb.edu/Faculty/Parhami/text_comp_arit.htm>.

Jan M. Rabaey, Anantha Chandrakasan, and Borivoje Nikolić, "Digital Integrated Circuits: A Design Perspective", 2nd Edition, Prentice-Hall, Upper Saddle River, NJ, 2003, pages 568, 575, 579, 590-593, 595-596.

Sunil P. Khatri and Narendra V. Shenoy, "Logic Synthesis", in EDA for IC Implementation, Circuit Design, and ProcessTechnology, Electronic Design Automation for Integrated Circuits Handbook series, Louis Scheffer, Luciano Lavagno, Grant Martin (Editors), CRC Press, Boca Raton, FL, 2006, pages 2-1 – 2-15.

Medardoni Simone, Bertozzi Davide, and Macii Enrico, "Power-Optimal RTL Arithmetic Unit Soft-Macro Selection Strategy for Leakage-Sensitive Technologies", Proceedings of the International Symposium on Low Power Electronics and Design, August 27–29, 2007, Portland, OR, pages 159-164.

Paul F. Stelling, Charles U. Martel, Vojin Oklobdzija, and R Ravi, "Optimal Circuits for Parallel Multipliers", IEEE Transactions on Computers, vol. 47, no. 3, March, 1998, pages 273-285.

Synopsys, "Design Compiler Reference Manual: Optimization and Timing Analysis", Mountain View, CA, October, 1999, pages 9-6 – 9-7, 9-26, 9-32, 9-27, 12-10 – 12-11, A-

1, A-4, A-11, A-21, F-5 – F-6.

Synopsys [Online], “Synopsys and NVIDIA Graphics”, Mountain View, CA, 2001, pages 1-4; last viewed December 13, 2007, at <http://www.synopsys.com/products/success/nvidia_ss_A4.pdf>.

Synopsys, “Design Compiler User Guide”, Mountain View, CA, June, 2002, page 8-36.

Synopsys [Online], “DesignWare® IP Family Reference Guide”, Mountain View, CA, March, 2007; last viewed December 13, 2007, at <http://www.synopsys.com/products/designware/docs/doc/dwf/manuals/dw_qrg.pdf>.

Junhyung Um and Taewhan Kim, “An Optimal Allocation of Carry-Save-Adders in Arithmetic Circuits”, IEEE Transactions on Computers, March, 2001, vol. 50, no. 3, pages 215-233.

Neil H. E. Weste and David Harris, “CMOS VLSI Design: A Circuits and Systems Perspective”, 3rd Edition, Pearson Education, Boston, MA, 2005, pages 166-182, 428-438, 662-663, 692-693, 696-702.