

Multi-Parameter Power Minimization of Synthesized Datapaths

W. Rhett Davis, Ambarish M. Sule, and Hao Hua
Department of Electrical and Computer Engineering
North Carolina State University
{rhett_davis, amsule, hhua}@ncsu.edu

Abstract

As processing technology continues to evolve, power minimization becomes more complex and crucial. Emerging technologies offer an array of different threshold voltages and gate oxide thicknesses. Along with choices of supply-voltage, parallelism, and pipelining, these options complicate the search for energy-optimal architectures. This paper explores the possibility of using convex optimization to solve the multi-parameter optimization problem and presents a case-study of an 8-bit multiply-accumulate block, which is optimized in 250nm and 70nm technologies.

1. Introduction

For a large part of the ASIC design community, the priority has shifted in recent years from timing optimization to power optimization. Embedded communication devices, which comprise an increasing portion of the semiconductor industry, typically adhere to standards that specify a minimum speed for a given circuit. In this environment, the main goal of the ASIC designer is to build the circuit that achieves the specified performance with the minimum power/energy possible.

As processing technology continues to evolve, power minimization becomes more complex and crucial. If current industry trends continue, leakage power in high-speed systems is expected to become more significant than active power for feature sizes around 50nm [1]. In order to balance the needs of high-performance and low-power designs, emerging technologies offer an array of different threshold voltages and gate oxide thicknesses. With so many options, simple architectural decisions such as the amount of parallelism or pipelining in a datapath can have a large effect on the energy efficiency of a system.

The task of power minimization of a synthesized datapath with so many process options is puzzling and difficult. Design tools such as Watt WatcherTM[2] and Power CompilerTM[3] offer ASIC designers a means to estimate and minimize power for a given standard-cell

library once a register-transfer-level (RTL) description for the circuit exists. In order to correctly identify the power-minimal architecture from a range of possibilities, designers must author RTL descriptions for each architecture and compare them using a range of standard-cell libraries. This process can be very time-consuming and raises the question of whether or not it is worth the effort.

This paper explores the question of how to minimize the power-consumption of a synthesized datapath. We begin by providing an overview of the power-minimization techniques that have inspired this work, focusing on balancing the energy-delay sensitivities of various design parameters. Next, we present our framework for optimization, which involves choosing the best parameters for a synthesized co-processor in a system-on-a-chip (SoC). Finally, we present our experimental results from the optimization of a multiply-accumulate block in 250nm and 70nm processes.

2. Overview of minimization techniques

A common technique for improving energy efficiency is supply-voltage scaling, which was presented by Chandrakasan et al. [4] and has been used in a number of experimental low-power system designs [5,6]. Subsequent work by Gonzalez et al. [7] explored the possibility of an optimal choice of both supply-voltage and threshold voltage to minimize energy-delay product (EDP). But as Zyuban and Kogge noted in [8], a minimum EDP circuit does not necessarily give the lowest-power circuit. Energy optimality can only truly be determined in terms of a specific deadline for computation.

Later work by Zyuban and Strenski [9] and Brodersen et. al [10], put forth the metric of hardware-intensity as a way to determine the energy-optimality of a system. The hardware-intensity θ of a given design parameter X is defined as

$$\theta(X) = -\frac{D}{E} \cdot \frac{\partial E / \partial x}{\partial D / \partial x} \Bigg|_{x=X} \quad (1)$$

The hardware-intensity can be considered the energy-delay sensitivity of a parameter, or the percent change in energy (E) per percent change in delay (D) for the parameter (x) evaluated for the current circuit. Formulating the problem in this way makes it possible to determine the optimum design for any number of parameters by the technique of convex optimization [11], as was used to determine optimal transistor sizes for static CMOS gates by TILOS [12]. The global minimum can be found when the θ values are zero. The minimum energy to satisfy a certain delay constraint can be found when the θ values are balanced (equal).

In order for this approach to work, it must be shown that as each optimization parameter is varied, the resulting energy-delay curve is convex. Otherwise, there is the possibility for the optimization technique to get trapped in a local minimum, which is not necessarily globally optimal.

Brodersen et. al [10] found convex, closed-form expressions for the hardware-intensity of supply voltage and threshold voltage, demonstrating that this method could be used to minimize the power-dissipation of a variety of simple circuits. Our goal for this work was to determine if certain architectural parameters exist that could be similarly used to optimize a complex, synthesized datapath. The following section describes how we parameterized our datapath according to parallelism, pipelining, and synthesis effort, in addition to the circuit parameters.

3. Optimization framework

In order to more thoroughly explore the issue of micro-architectural optimization, we revisit the issue of parallelism vs. pipelining. This example was examined by Chandrakasan et. al [4] to illustrate voltage scaling and was elaborated by Brodersen et al. [10] to illustrate multi-parameter power minimization. Fig. 1 shows how a micro-architecture can be generalized in terms of the number of pipeline stages (P) and the parallelism or radix (R). The initiation-interval (II), which is the time between the initiation of new computations, is shown, along with the delay (D) to complete N operations. Fig. 1(a) shows a reference function-unit to implement some function f , which is assumed to consist of combinational logic only. Fig. 1(b) shows the function-unit broken into a P-stage pipeline, which can be done automatically by a synthesis tool. In our case, we used the “optimize_registers” command in *Synopsys* Design Compiler™[3]. Fig. 1(c) shows R parallel function-units, each triggered by clock-phases offset by T_{cyc}/R , where T_{cyc} is the minimum cycle-time of the function unit. Fig. 1(d) shows how parallelism and pipelining can be combined. In this case, the clock phases are still

offset by T_{cyc}/R , where T_{cyc} is now the cycle-time of the pipeline.

The distinction between initiation interval and delay is important, because true energy optimality can only be determined in terms of the delay to compute a certain number of operations. The voltage-scaling arguments in [4] are focused on lowering average energy to get the same II. For values of N much larger than P and R, D converges to $N \cdot T_{cyc}$ for all four architectures, and optimizing energy in terms of II will be roughly

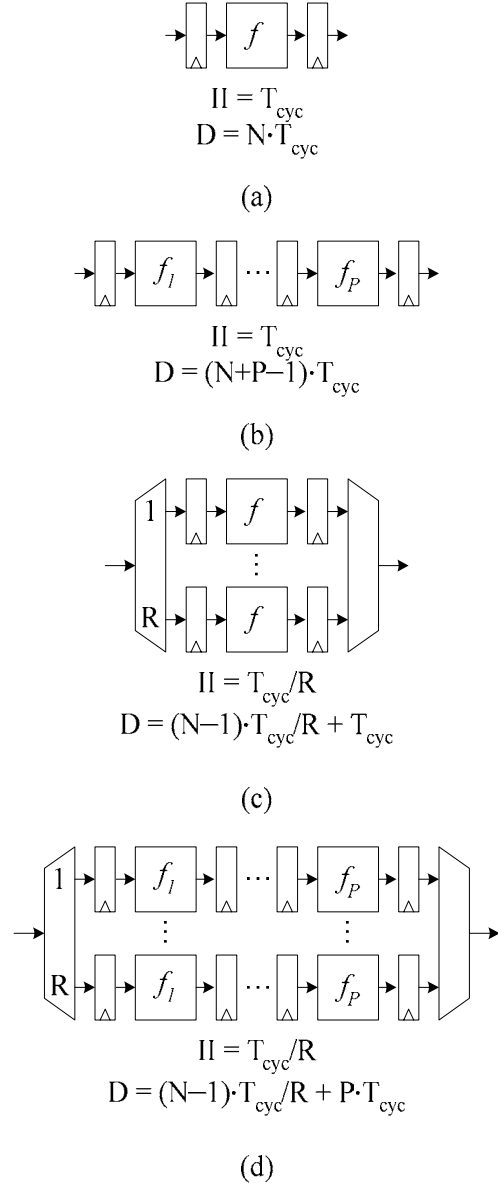


Figure 1: Micro-architecture example:
(a) reference, (b) pipelined design,
(c) parallel design, (d) parallel-pipelined design.

equivalent. For small values of N , however, the latency from pipelining or parallelism can give a very different optimum.

The complete micro-architecture as given in Fig. 1 is assumed to be a co-processor to a programmable processor in an SoC. While the programmable processor is assumed to run at the nominal supply-voltage for the technology, the co-processor is assumed to be in a separate supply-voltage “island” as presented in [13]. Level-shifters are needed to step signals up from a low-voltage to a higher-voltage. We used the level shifters in [13] and found that they contributed a negligible amount to the power but added significantly to the delay. This delay can be assumed to be pipelined, however, which means that there will be a significant increase in D for small values of N , but a negligible increase for large values of N .

In addition to the micro-architectural parameters of pipelining (P), parallelism (R), and supply-voltage (V_{DD}), we need to account for the range in architectures produced by the synthesis tool as it attempts to meet different timing constraints. To this end, we introduced the “synthesis-target” parameter (S), which represents the value of the cycle-time constraint in relation to the slowest and fastest circuits that the synthesis tool produces.

$$T_{cyc} = T_{min} + S \cdot (T_{max} - T_{min}) \quad (2)$$

T_{min} is found by setting the cycle-time constraint to zero and finding the negative slack in the result. T_{max} is found with a very-large, effectively infinite constraint. As will be shown in the next section, this parameter did not produce the convex energy-delay surface that we had hoped, but its effect on the energy-optimal architecture is too significant to be ignored.

The circuit chosen for comparison was an 8-bit multiply-accumulate, as shown in Fig. 2. For values of P larger than 1, pipeline registers were inserted between the adder and multiplier for retiming. The adder ultimately limits the reduction of T_{cyc} due to the feedback path, which cannot be retimed without changing the behavior. We chose this circuit because it contains many of the complexities of practical datapaths while still being relatively simple.

Based on the parameters P , R , V_{DD} , and S , we found the optimum architecture by exhaustive search, synthesizing a range of circuits and simulating them in HSPICE to find the energy. We used a 36-cell standard-cell library for the MOSIS SCMOS-DEEP rules in a 250nm technology. Designs were synthesized for values of P ranging from 1 stage to 5 stages and S ranging from 0 (fastest) to 1 (slowest). T_{cyc} was found using PathMillTM[3] with a set of model files for V_{DD} ranging

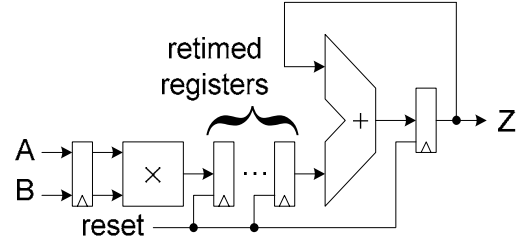


Figure 2: Optimized function-unit.

from 1.0V to 3.5V. D and I were found using the equations in Fig. 1, while the energy was found in HSPICE, averaging over 100 operations and using the cycle-times determined with PathMillTM. The effects of parallelism and level-shifters were added afterwards. The nominal supply-voltage for the system was assumed to be 2.5V, meaning that input level-shifters were used for V_{DD} above 2.5V, while output level-shifters were used for V_{DD} below 2.5V.

In order to expand our study to include the effects of multiple threshold-voltages and gate-oxide thicknesses, we developed HSPICE models for a 70nm technology based on the Berkeley Predictive Technology Model [14,15]. To mimic the trend in the latest processes, we chose three threshold-voltages based on published values [16,17] that are intended for high-speed, low-power, and general-purpose designs. Values for L_{eff} , well resistances, and two physical gate-oxide thicknesses (7Å and 12Å) were taken from ITRS predictions [18], adjusting them by 4Å to account for charge layer quantization and polygate depletion. Table 1 summarizes the technology model.

Table 1: Predictive 70nm technology values

		NMOS	PMOS
V_t	Low Power (LP)	0.26 V	-0.30 V
	General Purpose (G)	0.18 V	-0.25 V
	High Speed (HS)	0.10 V	-0.21 V
t_{ox}	Thick	16 Å	16 Å
	Thin	11 Å	11 Å

* $L_{eff} = 40$ nm and $R_{sdw} = 170$ Ω in all cases.

4. Convexity of energy-delay curves

In this section, we explore the question of whether or not the parameterized, multi-dimensional energy-delay surface is convex. A multi-dimensional surface is convex if any straight line-segment with endpoints on the surface does not intersect the surface at any other point [11]. For some parameters, such as supply voltage, the surface is obviously convex. Fig. 3 shows a subset of the surface created by varying V_{DD} in the 250nm technology while holding the other parameters constant.

Similar convex curves were noted for all other parameter choices. We also note that the curves generated from varying the parallelism are convex, since increasing R reduces the delay and increases energy linearly, except for a small effect from the input DEMUX and output MUX logic.

The parameters that affect the synthesis tool, however, do not show convex energy-delay surfaces. Fig. 4 shows the energy-delay values for all synthesized circuits with the same supply voltage. The graph shows that for $P=1$, the cycle-time goal set by S is met by the synthesis tool. As P increases, however, the design is more tightly constrained, and the resulting cycle-time is no longer correlated with S . When we consider energy, we see that there is as much as a 20% difference in energy from varying S , but there is no value of S that consistently gives the lowest energy. If we want to use the parameters of P and S for convex optimization, then we need to reduce set of possible circuits to a pareto-optimal set. As illustrated in the figure, the pareto-optimal set is ordered so that it gives the minimum

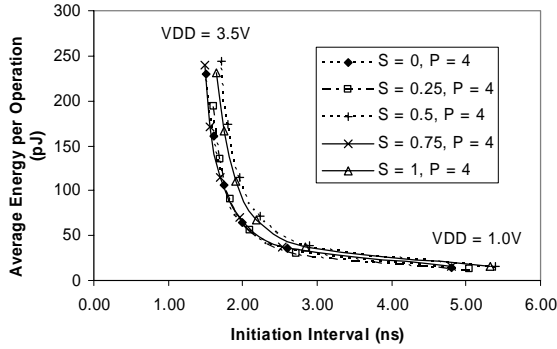


Figure 3: Effect of varying V_{DD} in the 250nm technology, for $P=4$ and $R=1$.

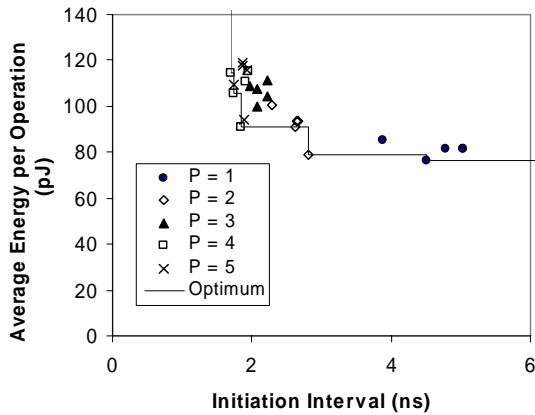


Figure 4: Effect of varying P and S in the 250nm technology, for $V_{DD}=2.5V$ and $R=1$.

energy for a certain maximum delay. We note from the figure that the pareto-optimal set for this experiment includes pipeline stages of 1,2 and 4 only. In spite of the added register and clock energy for the fourth pipeline stage, glitching was reduced sufficiently to bring the average energy below that of the 3-stage pipeline. By arriving at the pareto-optimal set of micro-architectures we can continue to consider the problem in terms of convex-optimization.

For the additional parameters in the 70nm technology, we notice convexity for t_{ox} but not V_t . Fig. 5 shows a subset of the t_{ox} variation. For these and all other parameter choices, the thin oxide always produced a faster design that used more energy, primarily from the increase in gate capacitance as well as gate leakage-currents. The variation with threshold-voltage, however, was not as consistent. Fig. 6(a) shows a subset of curves that illustrate the effect we would normally expect: the energy-delay surface is convex, with energy increasing

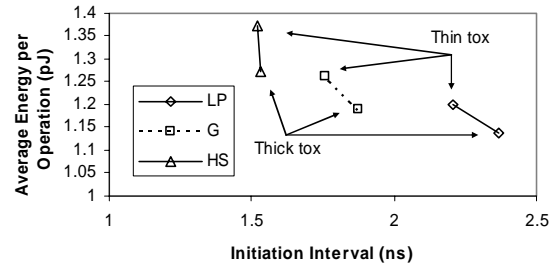
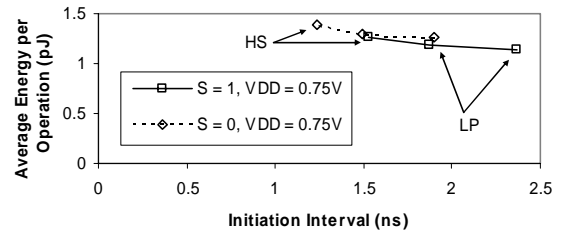
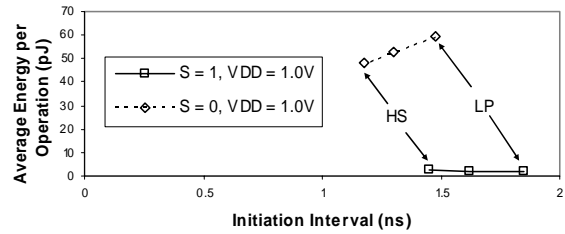


Figure 5: Effect of varying t_{ox} in the 70nm technology, for $V_{DD}=0.75V$ and $R=1$.



(a)



(b)

Figure 6: Effect of varying V_t in the 70nm technology, for $P=1$, $R=1$, and $t_{ox}=\text{thick}$.

as V_t drops due to increased leakage current. Fig. 6(b), however, shows that for certain very aggressively optimized circuits, the highest threshold can yield the most energy dissipation for high values of V_{DD} . This is due to increased short-circuit currents during the longer transition times with the higher threshold. The change in direction amounts to a “twist” in the multi-dimensional energy-delay surface that makes it inappropriate for convex optimization. This contradicts the finding in [10], which predicted that V_t variation would be convex. However, our data suggests that the surface is convex if we restrict ourselves to values of V_{DD} at or below the nominal supply-voltage of 0.75V.

5. Optimization results

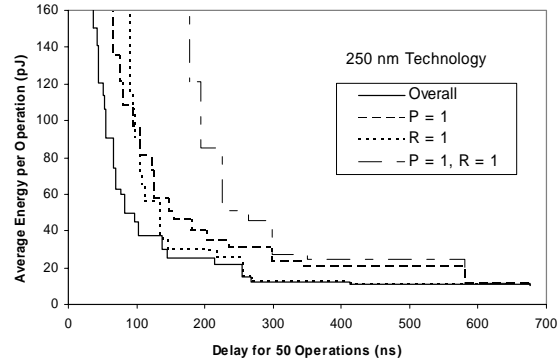
Here we present the results of the multi-parameter exhaustive search for the minimum-power architecture. Each plot is shown as a pareto-optimal energy-delay curve for some subset of the search-space. The delay for 50 operations was chosen because it was sufficiently large compared to R and P to be considered as infinite. For smaller numbers of operations, the results will be similar but will favor less pipelining and parallelism.

First, consider the issue of parallelism vs. pipelining. Fig. 7(a) shows the optimal energy-delay curve in 250nm when searching over all parameters. The R=1 curve shows the result when parallelism is not allowed. For delays above 250ns, this curve merges with the overall optimal curve, while the P=1 curve is nearly twice as high. This means that pipelining is more energy-efficient than parallelism for long delay constraints. As the delay constraint is brought down, however, parallelism becomes relatively more efficient until the curves cross at around 100ns. This is due to the fact that higher and higher supply voltages are needed for the pipelined circuit to match the performance of the parallel circuit. Most interesting, however, is the fact that the combination of parallelism and pipelining provides an energy savings of roughly 50% over either one at 100ns.

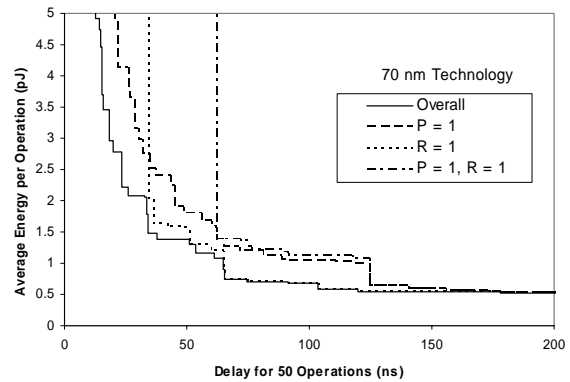
Fig. 7(b) shows the same analysis in 70nm. The primary difference between this graph and the one in Fig. 7(a) is the R=1 curve doesn't cross the P=1 curve until around 35ns, which is the absolute maximum speed for the pipelined design. This supports the findings in [10], which show that pipelining becomes increasingly more energy-efficient than parallelism as devices become leakier. Just as with the 250nm case, however, the combination of parallelism and pipelining is able to produce much more energy-efficient circuits for at faster speeds than either parallelism or pipelining alone.

Figs. 8-12 show a more detailed picture of the effect of each parameter for the 70nm technology. Fig. 8

shows that the high-speed threshold voltage is the energy-optimal threshold for delays below 180ns.



(a)



(b)

Figure 7: Optimal energy-delay curves comparing pipelining and parallelism (a) in 250nm and (b) in 70nm.

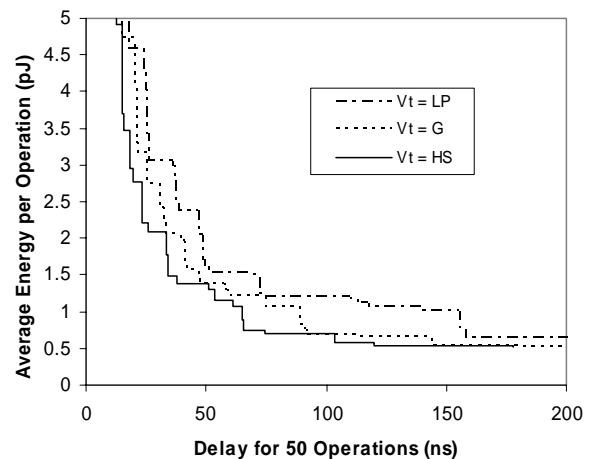


Figure 8: Optimal energy-delay curves comparing threshold voltages.

Although not shown in the figure, the general purpose threshold was energy-optimal for delays ranging from 180 to 450ns, while the low-power threshold was optimal for longer delays. This is due to the relative importance of leakage to dynamic power for the different delay constraints. Fig. 9 shows that the effect of changing oxide thickness is small. The curves are very close and wind around each other, showing no clear trend. The supply voltage trend in Fig. 10 is much more significant, showing that the minimum energy is achieved at the minimum supply voltage in almost all cases.

Figs. 11 and 12 illustrate the trade-off between different degrees of parallelism and pipelining. Consistent with Fig. 7(b), Fig. 11 shows that increasing

degrees of parallelism become optimal as the speed requirement increases. At around 65ns, parallelism of 2 become optimal (not shown), at 35ns, parallelism of 3, and so on. The increase in pipelining in Fig. 12 tracks the improvement in delay shown earlier in Fig. 4, increasing from 1 pipeline stage to 2 and then 4 as the delay decreases. Then, for a delay of around 65ns, the parallelism rises to 2, and the optimum cycles again from 1 to 2 to 4.

Based on the data, we can conclude an approach to manually optimize a high-throughput system. Except for the most relaxed of speed requirements, the lowest energy architecture is consistently the one with the lowest-threshold voltage (HS) and lowest supply voltage. The next choice appears to be the parallelism,

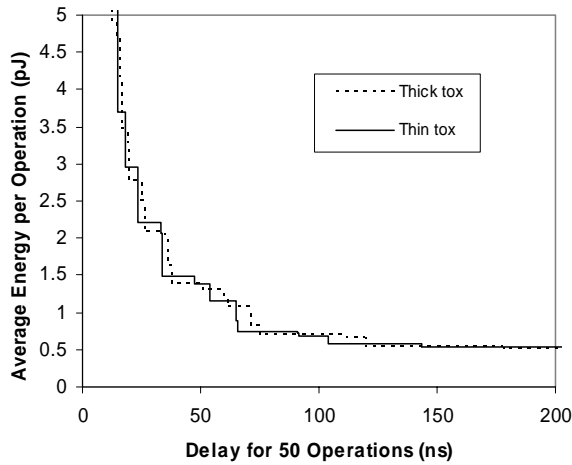


Figure 9: Optimal energy-delay curves comparing oxide thickness.

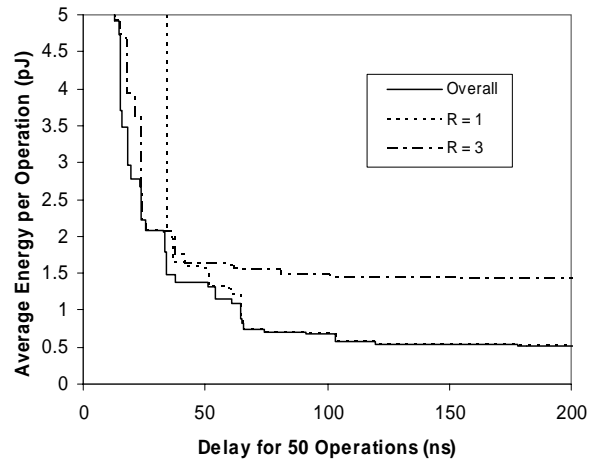


Figure 11: Optimal energy-delay curves comparing parallelism.

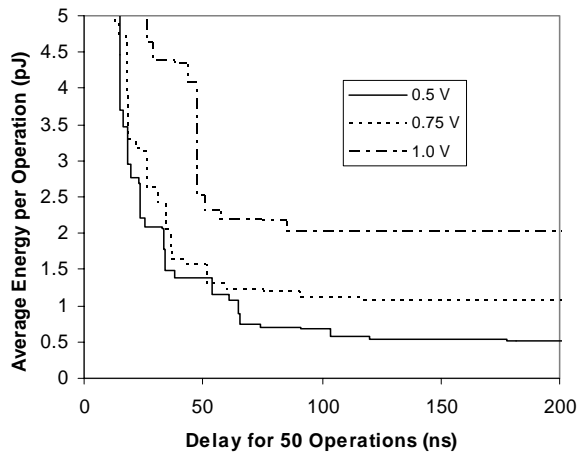


Figure 10: Optimal energy-delay curves comparing supply voltage.

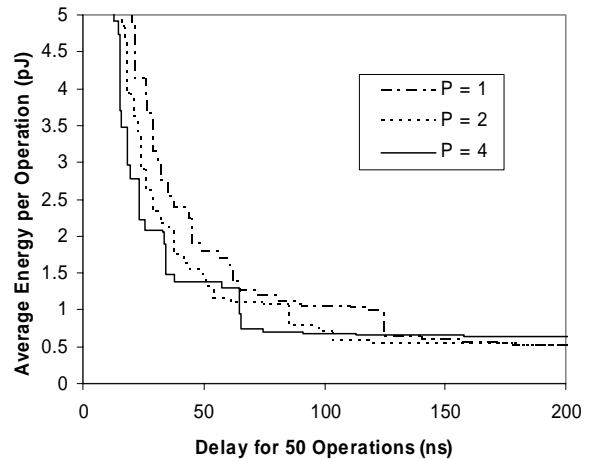


Figure 12: Optimal energy-delay curves comparing pipeline stages.

which should be kept as low as possible to avoid leakage currents. The last choices are the pipeline depth and oxide thickness, which can be tuned to get the lowest energy for the desired throughput.

6. Conclusions

We have presented data to support the idea of using convex optimization to minimize the power dissipation of synthesized datapaths for maximum computation-time constraints. Two of the key obstacles are the “twist” in the threshold-voltage dimension of the energy-delay surface, which can be avoided by using lower supply voltages, and the ordering of architectures according to their pareto-optimality. The potential value of solving these issues lies in their application to system-level synthesis tools. Virtually all of the choices made in the examples presented here must typically be made at the beginning of the hardware design-process, before the RTL description has been authored and before accurate information about power-performance is known. The goal of system-level hardware-design tools should be to generate hardware that is verifiably close to the power-minimal solution, or at least the lowest-power solution that a circuit-designer is likely to uncover.

References

- [1] S. Stiffler, “Optimizing Performance and Power for 130 Nanometer and Beyond,” *IBM Technology Group New England Design Forum*, June 18, 2003.
- [2] Watt Watcher™ from *Sequence Design, Inc.*, <http://www.sequencedesign.com>.
- [3] Design Compiler™, Power Compiler™, and Path Mill™ from *Synopsys, Inc.*, <http://www.synopsys.com>.
- [4] A. P. Chandrakasan and R. W. Brodersen, “Low-power CMOS Digital Design,” *IEEE JSSC*, Apr. 1992, pp. 473-84.
- [5] T. Burd et al., “A Dynamic Voltage Scaled Microprocessor System,” *ISSCC Dig. Tech. Papers*, Feb. 2000, pp. 294-5.
- [6] T. Kuroda et al., “Variable Supply-Voltage Scheme for Low-Power High-Speed CMOS Digital Design,” *IEEE JSSC*, March 1998, pp. 454-62.
- [7] R. Gonzales, B. Gordon, and M. A. Horowitz, “Supply and Threshold Voltage Scaling for Low-Power CMOS,” *IEEE JSSC*, Aug. 1997, pp. 1210-6.
- [8] V. Zyuban and P. Kogge, “Optimization of High-Performance Superscalar Architectures for Energy Efficiency,” *Proc. ISLPED*, Jul. 2000, pp. 84-9.
- [9] V. Zyuban and P. Strenski, “Unified Methodology for Resolving Power-Performance Tradeoffs at the Microarchitectural and Circuit Levels,” *Proc. ISLPED*, Aug. 2002, pp. 166-71.
- [10] R. W. Brodersen, M. A. Horowitz, B. Nikolic, and V. Stojanovic, “Methods for True Power Minimization,” *Proc. ICCAD*, Nov. 2002, pp. 35-42.
- [11] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, 1970.
- [12] J. P. Fishburn and A. E. Dunlop, “TILOS: A Posynomial Programming Approach to Transistor Sizing,” *Proc. ICCAD*, Nov. 1985, pp. 326-328.
- [13] R. Puri et al., “Pushing ASIC Performance in a Power Envelope,” *Proc. DAC*, Jun. 2003, pp. 788-93.
- [14] C. Hu, “BSIM Model for Circuit Design Using Advanced Technologies,” *Symp. VLSI Circuits Dig. Tech. Papers*, June 2001, pp. 5-10.
- [15] The Berkeley Predictive Technology Model, <http://www-device.eecs.berkeley.edu/~ptm>.
- [16] C. C. Wu et al., “A 90-nm CMOS Device Technology with High-speed, General-purpose, and Low-leakage Transistors for System on Chip Applications,” *IEDM Tech. Dig.*, Dec. 2002, pp. 65-8.
- [17] S. Tyagi et al., “A 130 nm Generation Logic Technology Featuring 70 nm Transistors, Dual Vt Transistors and 6 layers of Cu Interconnects,” *IEDM Tech. Dig.*, Dec. 2000, pp. 567-70.
- [18] *The International Technology Roadmap for Semiconductors*, <http://public.itrs.net>.