Limit Study of Energy & Delay Benefits of Component-Specific Routing

Nikil Mehta
Department of Computer Science, 256-80
California Institute of Technology
Pasadena, CA 91125
nikil@caltech.edu

Raphael Rubin
Department of Computer and Information Systems
University of Pennsylvania
3330 Walnut Street
Philadelphia, PA 19104
rafi@seas.upenn.edu

André DeHon
Department of Electrical and Systems Engineering
University of Pennsylvania
200 S. 33rd St.
Philadelphia, PA 19104
andre@acm.org

ABSTRACT

As feature sizes scale toward atomic limits, parameter variation continues to increase, leading to increased margins in both delay and energy. The possibility of very slow devices on critical paths forces designers to increase transistor sizes, reduce clock speed and operate at higher voltages than desired in order to meet timing. With post-fabrication configurability, FPGAs have the opportunity to use slow devices on non-critical paths while selecting fast devices for critical paths. To understand the potential benefit we might gain from component-specific mapping, we quantify the margins associated with parameter variation in FPGAs over a wide range of predictive technologies (45nm–12nm) and gate sizes and show how these margins can be significantly reduced by delay-aware, component-specific routing. For the Toronto 20 benchmark set, we show that component-specific routing can eliminate delay margins induced by variation and reduce energy for energy minimal designs by 1.42–1.98×. We further show that these benefits increase as technology scales.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids—*placement and routing*; B.8.1 [**Performance and Reliability**]: Reliability, Testing, and Fault-Tolerance

General Terms

Algorithms, Measurement, Reliability

Keywords

Component-Specific Mapping, Variation Tolerance, Minimum Energy

1. INTRODUCTION

As we continue to scale down feature sizes, we can no longer guarantee that all transistors will have identical parameters, nor can we determine prior to fabrication which transistors will be fast, slow, or defective. This parameter uncertainty leads to circuits that are pessimistically sized, underclocked, and overvoltaged. Because designers have no guarantee that critical paths will not contain any slow transistors, they must accept lower frequency operation. As a corollary, to maintain cycle time, V_{dd} may need to be increased in order to speed up slow transistors to meet timing, wasting energy. This energy loss is particularly detrimental for many modern designs that are energy limited. Static delay and energy margins are already significant in current technologies and will only continue to increase.

Conventional variation tolerance techniques will not be effective at reducing these margins in future technology nodes that are dominated by large random V_{th} variations [1]. For the 22nm node the ITRS predicts[1] $\sigma_{V_{th}}/\mu_{V_{th}} = 27\%$, meaning that a minimum sized transistor (L=W=22nm) with a nominal $V_{th}{\approx}300$mV can have a 3σ V_{th} spread of 57mV–543mV, equating to a delay difference of 0.49–12.58 ps (Fig. 2), or nearly two orders of magnitude. The random, fine-grained nature of this delay spread means that conventional techniques that rely on pre-fabrication estimates (e.g. SSTA, statistical static timing analysis) and coarse-grained adaptations (e.g. adaptive body biasing) can only be of limited use. Sizing up transistors is an important conventional technique to reduce variation, but this comes with a cost of increased capacitance and energy—a tradeoff that must be quantified under variation (see Sec. 5.3).

FPGAs provide a unique opportunity for fine-grained control and post-fabrication configurability. If we break the one-mapping-fits-all approach and map on a per-chip basis, tools can place and route designs that avoid unreasonably slow resources while strategically selecting fast resources for critical paths. Unfortunately, there are high barriers to component-specific mapping. Individual resource delays must be extracted on a per-chip basis. Mapping must also be performed per-chip, potentially exploding CAD effort. While these challenges are great, recent work (see Sec. 2.2.1 and 2.2.2) demonstrates promising progress.

Nonetheless, the question remains as to what are the ultimate achievable benefits from component-specific mapping.

[1]ITRS Table DESN9 in Design Section reports 3σ variation (81% for 22nm), which we divided by 3.

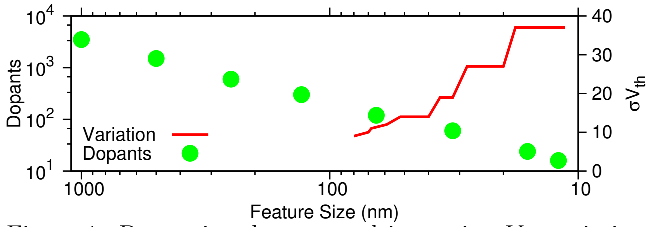

Figure 1: Decreasing dopants and increasing V_{th} variation from ITRS 2010

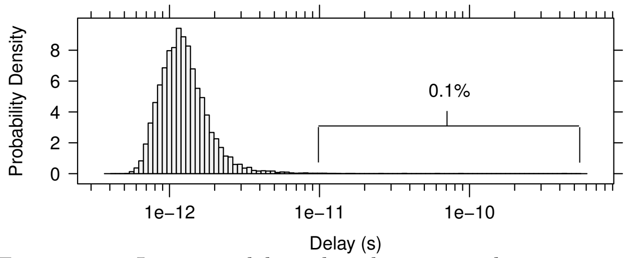

Figure 2: Inverter delay distribution under variation (W_p=W_n=L=22nm)

To answer this question we perform a limit study of the delay and energy benefits from component-specific mapping for future technology nodes. As a limit study, we make optimistic assumptions about the completeness and precision of delay information available and about the amount of effort we can apply to routing. We examine a router that assumes knowledge of all FPGA resource delays and measure the routed delay and energy of the Toronto 20 benchmarks [4] for predictive technology models [36] spanning the 45nm, 32nm, 22nm, 16nm, and 12nm nodes. We characterize the expected delay and energy margins from variation and by how much we expect to reduce these margins through component-specific mapping. We also explore how gate sizing affects these margins and expected benefits.

Novel contributions of this work include:

- Quantification of FPGA delay and energy margins due to random V_{th} variation in interconnect for 45nm–12nm technology nodes.
- Quantification of potential delay and energy savings for component-specific routing.
- Demonstration of the impact of interconnect buffer sizing on margins and delay and energy savings.
- Demonstration of the impact of CLB IO sparing.
- Determination of energy-optimal sizing as a function of technology and variation.

2. BACKGROUND

2.1 Variation, Delay & Energy Scaling

Parameter variation can be decomposed into lot-to-lot, wafer-to-wafer, die-to-die and within die (WID) variations. WID variation is the most significant contributer to parameter uncertainty, and can be further categorized as systematic (e.g. layout dependent), spatially correlated (e.g. distance dependent), and random. Random WID V_{th} variation, caused by effects like dopant fluctuation and channel length variance, is expected to dominate future sources of variation [1,3]. Fig. 1 shows how V_{th} variation increases due to decreases in dopant count as a function of feature size.

Increased Delay.

The first impact of variation is that gates will exhibit a large spread in delays, reducing the speed of designs as the delay of a circuit is set by its slowest path. We can express the current and delay of a gate as:

$$I_{sat} = W v_{sat} C_{ox} \left(V_{gs} - V_{th} - \frac{V_{d,sat}}{2} \right)^{\gamma} \quad (1)$$

$$I_{sub} = \frac{W}{L} \mu C_{ox} (n-1) (v_T)^2 e^{\frac{V_{gs}-V_{th}}{n v_T}} \left(1 - e^{\frac{-V_{ds}}{v_T}} \right) \quad (2)$$

$$\tau_p = \frac{C V_{dd}}{I_{on}} \quad (3)$$

Where I_{on}=I_{sat} for $V_{dd} \geq V_{th}$, and I_{on}=I_{sub} for $V_{dd} < V_{th}$. V_{th} variation can cause a large spread in gate delays: the HSPICE simulated delay of a minimum sized 22nm inverter (W=L=22nm) with $\sigma_{V_{th}}/\mu_{V_{th}}$=27% and 10,000 samples is shown in Fig. 2, and we can see delays that span several orders of magnitude. Because of Fig. 1, this spread will increase with continued technology scaling.

Increased Energy.

The second impact of variation is that it raises energy per operation, which can be expressed (ignoring short circuit currents and glitches) as follows:

$$E_{dynamic} = \sum_{i}^{n_{switch}} \frac{\alpha_i}{2} \cdot C_{load,i} \cdot (V_{dd})^2 \quad (4)$$

$$E_{static} = \sum_{i}^{n_{total}} I_{leak,i} \cdot V_{dd} \cdot \tau_p \quad (5)$$

$$E_{total} = E_{dynamic} + E_{static} \quad (6)$$

For a circuit operating at a target delay, to compensate for slower gates, designers are often forced to raise V_{dd} to increase transistor drive strength (I_{sat}, Eq. 1), increasing energy/operation.

Alternatively, many circuits may need to simply minimize energy/operation at the cost of delay. Energy/operation is the primary design constraint for many low power, embedded systems. It correlates directly with battery lifetime, and is also relevant to operating with limited thermal budgets. Even for these delay unconstrained systems, parameter variation makes energy minimization more difficult [6]. From Eq. 6 we see that the most direct way to reduce energy/operation is to lower V_{dd}. For most circuits, $E_{dynamic} > E_{static}$, so voltage scaling yields a quadratic reduction in energy. This technique has been examined in commercial FPGAs [9].

With $V_{dd} > V_{th}$ the delay of a transistor depends on Eq. 1 and is super linear; however, when $V_{dd} < V_{th}$ delay solely depends on Eq. 2 making it exponential in both V_{dd} and V_{th}. The exponential dependence in V_{dd} means that operations become significantly longer at lower voltages. As static energy/operation is expressed as leakage power times the length of an operation (Eq. 5), at low V_{dd} it will increase dramatically and eventually become the dominant source of energy dissipation. Fig. 3 shows the energy/operation of a 16-bit multiplier mapped to a 22nm FPGA. As a result, *there exists a V_{dd} at which energy is minimized.* This gives a well-defined target point of operation—we should operate at the energy optimal V_{dd} when minimizing energy/oper-

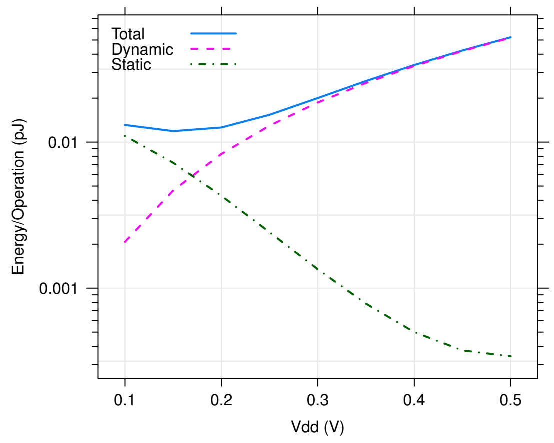

Figure 3: Minimum energy/operation for a 16-bit multiplier at 22nm (V_{th}=300mV)

ation. Furthermore, note that the minimum energy point occurs below the threshold voltage in this example, as it does for most designs. If we add V_{th} variation, the length of an operation will increase further due to delay uncertainty, increasing the minimum energy of operation [7].

Increased Failures.

The third impact of parameter variations is the increase in functional failures due to V_{th} mismatch [26]. SRAMs are commonly the first circuits to fail due to V_{th} variation which causes read upsets, write upsets, hold failures, and access time failures. However, FPGA configuration SRAMs simply hold state and are not cycled during operation, so these circuits can be fabricated with increased widths and with a separate, high V_{th} to avoid failures [34].

Static logic, however, can also fail due to V_{th} variation. We define a CMOS inverter to be defective due to variation when leakage current overpowers on current:

$$I_{PMOS,off} > I_{NMOS,on} \text{ or } I_{PMOS,on} < I_{NMOS,off} \quad (7)$$

Under these conditions the inverter can never switch; this can only happen when V_{th} variation is large enough such that a very high V_{th} device is paired with a very low V_{th} device. Furthermore, as V_{dd} decreases, the probability of a defect increases as I_{on} for both PMOS and NMOS transistors degrades (Eq. 1) [16]. Consequently, this effect is particularly acute for subthreshold operation, preventing operation near the minimum energy point.

2.2 Prior Work

Component-specific mapping for FPGAs is a simple idea: if you can determine resource delays on a chip before mapping and use that information during mapping, then you can naturally produce a more efficient design. This allows the design to avoid defects (Eq. 7) and avoid slow gates on the critical path. What is not obvious is a) how to extract resource delays, b) how to efficiently use that information to route every chip differently, and c) what is the overall benefit. This paper only attempts to quantify the third: the benefits of component-specific mapping. The ultimate feasibility of component-specific mapping is still an open question. However, several of the key challenges in measurement and mapping have already been solved, representing partial, but realistic solutions for component-specific mapping.

2.2.1 *Component-Specific Measurement*

Several researchers have identified ways to quickly and accurately measure FPGA resource delays without the use of an expensive tester. One technique uses arrays of configured ring oscillators to measure aggregate delays of N=5–7 stages of LUT + interconnect for commercial 90nm and 65nm devices [29, 33]. Additionally, Wong et al. developed a technique for at-speed path delay measurements that can be configured to extract delays of N=2 stages of LUT + interconnect with 1 ps resolution [35]. The technique configures a path between a registered source LUT and a shadow registered destination LUT and sweeps clock frequency until the shadow register detects an error. Full characterization of a Altera Cyclone II EP2C35 can be achieved in 3 seconds. Using sparse sampling, Majzoobi et al. showed that spatially correlated variation can be approximated in milliseconds and characterized with only megabytes of data [23]. Remaining challenges in measurement are to resolve single element delays (e.g. individual LUTs, switches as opposed to aggregate paths) and to obtain even finer timing resolution.

2.2.2 *Component-Specific Mapping*

Initial work in component-specific mapping focused on defect tolerance, with HP's TERAMAC demonstrating the ability to locate and map around defective elements and tolerate defect rates of 3–10% [10]. Later ideas in defect-tolerant FPGA component-specific mapping generated multiple bitstreams and then tested each bitstream per component [24, 30, 32].

Recent work has performed mapping using delay knowledge in both placement and routing. Katsuki et al. [8] and Cheng et al. [13] perform chip-wise placement by generating a variation map per chip and using that map to place critical logic in fast regions assuming the dominant WID variation is spatially correlated. However, future technologies are dominated by random, not spatially correlated variation; placement is too coarse grained to preferentially select individual devices. Gojman et al. [11] perform fine-grained component-specific routing on an reconfigurable NanoPLA. By matching net fanout to threshold voltages, they are able to restore 100% yield in a 5nm technology with $\sigma_{V_{th}}/\mu_{V_{th}} = 38\%$.

The cost of mapping with component-specific techniques can be prohibitively expensive. Since recent FPGAs contain billions of transistors, measurement storage may be significant. More importantly, CAD must be performed per-chip. Modern CAD runtime for large designs often takes days; multiplying this CAD effort by the number of shipped parts explodes handling time.

One promising direction in reducing storage and CAD effort is Choose-Your-own-Adventure (CYA) routing [27]. In CYA, routing is performed once for all chips, but the bitstream produced by the router contains several alternative routes for each net in the design that are evaluated at load time. CYA could be conceivably extended to perform timing tests for measurement (Sec. 2.2.1) to enable delay-aware component-specific mapping.

2.2.3 *Conventional Variation Tolerance*

To tolerate FPGA parameter variation, researchers have employed many of the same techniques used for ASICs and

CPUs. Adaptive body biasing [25] and dual-V_{dd} assignment [5] attempt to compensate for variation at the CLB level by adjusting V_{th} and V_{dd} post fabrication. However, these techniques have insufficient granularity to deal with random variation due to circuity overhead.

Several researchers have examined using SSTA in the timing analysis steps of clustering, placement [15, 21], routing [31], and the entire CAD flow [19, 20] to better identify and optimize statistically critical paths under variation. Lin et al. demonstrate a mean delay improvement of 6.2% and a delay variance reduction of 7.5% for combined regional and random $\sigma_{V_{th}}/\mu_{V_{th}}$ of 2% and 3.3% respectively. SSTA relies on pre-fabrication models that are difficult to generate accurately, do not scale well for high V_{th} variation, and do not reflect the individual variation map of a given chip.

Device and circuit parameters can also be optimized to mitigate the impact of parameter variation. Transistor sizing is a common strategy used in ASICs to directly reduce the magnitude of variation. Larger transistors have reduced V_{th} variation as can be seen by the following relation:

$$\sigma_{V_{th}} \propto \frac{1}{\sqrt{WL}} \tag{8}$$

Increasing W in logic transistors can increase variation tolerance, at the cost of area and energy. Modern integrated circuits use few or no minimum size devices for this reason. We quantify the benefits and drawbacks of sizing both for one-mapping-fits-all and for component-specific mapping, using Eq. 8 to scale the variation of sized transistors.

3. ANALYSIS

Before looking at the results, this section briefly reviews the major effects at play to provide an intuitive basis for reasoning about the results and how they might be impacted by different designs and benchmarks.

Device-level variation does not linearly result in circuit- or application-level variation. In this section, we identify the major design and platform properties that couple with device-level variation to determine application-level performance. Note the models used in this section are simplistic, treating the primitive delays as Gaussian random variables—for the actual circuit modeling performed later in the paper, we use more primitives variables (*e.g.* $\sigma_{V_{th}}$) as the basis for computing delays.

3.1 Path Length

The set of device parameters will combine to define the delay of the each device, gate, or interconnect segment. For intuition, let us think about gates and interconnect segments as the unit of composition. Assume each gate or interconnect segment has delay τ_u, and further assume the delay of each unit is an identically and independently distributed (i.i.d.) random variable taken from a Gaussian distribution with mean μ_{τ_u} and standard deviation σ_{τ_u}.

Disregarding fanout and reconvergent paths, the delay along a path of length d is the sum of d gate delays. The sum of a set of Gaussians is, itself, a Gaussian.

$$\tau_{path}(d) = \tau_{u_0} + \tau_{u_1} + \ldots + \tau_{u_{d-1}} \tag{9}$$

$$\mu_{\tau_{path}(d)} = d \times \mu_{\tau_u} \quad ; \quad \sigma_{path}(d) = \sqrt{d} \times \sigma_{\tau_u} \tag{10}$$

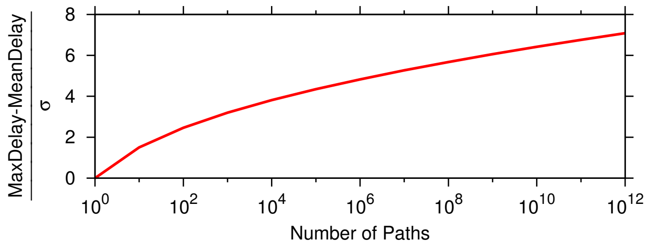

Figure 4: 50% Yield Delay vs. Number of Paths

Note that:

$$\frac{\sigma_{path}(d)}{\mu_{\tau_{path}(d)}} = \left(\frac{1}{\sqrt{d}}\right) \frac{\sigma_{\tau_u}}{\mu_{\tau_u}} \tag{11}$$

That is, as a percentage of the nominal delay, the variation decreases by a factor of the square root of the path length, d. *This means, slow circuits will see small variation, while circuits that are highly pipelined with few LUTs and interconnect segments between flops will see a larger variation as a percentage of nominal path delay.* Note that:

- It is these high-performance, short paths where we care about performance the most.
- We expect pipelining to increase in future designs.

3.2 Multiple Paths

Now, consider that we have K independent, parallel paths on a chip that have the same nominal delay; i.e. all are critical paths. Few paths on the chip are independent, but we make this simplifying assumption to develop intuitive analysis on the scaling trends. If we clock the chip synchronously, the clock cycle is limited by the longest path. This gives us:

$$T_{cycle} = \max_{\text{all paths } p_i} (\tau_{p_i}) \tag{12}$$

We would like to know the distribution of T_{cycle}, which is a max of Gaussians.

To simplify this from a distribution to a single number, we might ask what delay we can expect half of our components to meet, $T_{50\%}$. That means:

$$P(T_{cycle} \leq T_{50\%}) = 0.5 \tag{13}$$

For T_{cycle} to be $T_{50\%}$, then all K paths must have delay less than $T_{50\%}$.

$$P(T_{cycle} \leq T_{50\%}) = \left(P(\tau_{p_i} < T_{50\%})\right)^K = 0.5 \tag{14}$$

This tells us:

$$P(\tau_{p_i} < T_{50\%}) = (0.5)^{(1/K)} \tag{15}$$

When K is large, this means that $P(\tau_{p_i} < T_{50\%})$ must be a value very close to 1; for the Gaussian cumulative distribution function (Φ) to be close to one, we must allow the random variable, $T_{50\%}$, to be many σ above the mean.

$$N_\sigma(K) = \Phi^{-1}\left((0.5)^{(1/K)}\right) \tag{16}$$

This gives us:

$$T_{50\%} = \mu_{\tau_{p_i}} + N_\sigma(K) \times \sigma_{\tau_{p_i}} \tag{17}$$

Fig. 4 plots N_σ as a function of K.

This suggests: *the more critical paths we have, the slower we can expect the final circuit delay to be.* As we scale to larger ICs, and hence more paths, this effect will increase.

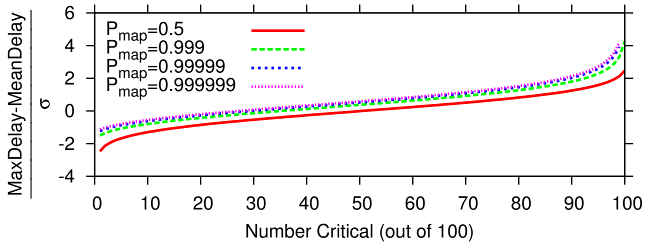

Figure 5: Achievable Delay vs. Near Critical Functions within a set of 100 Varying Resources

3.3 Choice

The path effect above results when we are forced to use every resource. We are forced to slow down the entire circuit because one or a few units are likely to be very slow. If we can choose devices to use, we can avoid these bad devices. For example, if the gates or segments are Gaussian distributed, then half of them are faster than the mean and half are slower. If we had a component with twice the resources of those needed (e.g. twice the channel width, twice the LUTs per CLB), we could avoid the half of the resources that are slower than the mean and guarantee that most ICs run at least as fast as the mean delay. When we have units with slack, the slack nodes can act as "extra" resources for the near critical resources. For example, when only 20% of the logical functions contending for resources in an interchangeable resource pool are near critical, these critical 20% effectively see an overpopulation ratio of 5.

More generally and formally, if we have equivalent sets of resource of size N and map to only use M of them, then the probability of yielding the M resources is:

$$P_{map} = \sum_{M \le i \le N} \left(\binom{N}{i} \left(P_u(\tau_{ref})\right)^i \left(1 - P_u(\tau_{ref})\right)^{N-i} \right)$$

Here, P_u is a Gaussian cumulative distribution function for τ_u. We define:

$$P_u(\tau_{ref}) = P\left(\tau_u \le \tau_{ref}\right) \qquad (18)$$

For fixed M and N, we can invert this and ask what P_u results in a given level of P_{map}. In turn, this tells us what τ_{ref} we can expect to achieve in order to meet the P_u bound. Fig. 5 shows speed achievable, as we vary the number of near critical units (M) contending for fast resources in a fixed pool of N=100 resources. *The more resources available for critical functions to choose from, the higher the performance we can achieve.*

3.4 Voltage for Fixed Timing Target

If, instead, we want to adjust the supply voltage (V_{dd}) to target a fixed timing target, the phenomena is similar to timing in Sec. 3.2, except the overhead σ is now in voltage. For simplicity, we assume we have K transistors that are critical and ignore the non-critical transistors. If we know we can achieve the desired timing with the nominal voltage $\mu_{V_{dd}}$, giving us a transistor on-current (I_{sat}, Eq. 1) for $\mu_{V_{on}} = \mu_{V_{dd}} - \mu_{V_{th}} - \frac{V_{d,sat}}{2}$, then we need to supply the chip with a V_{dd} large enough to provide this drive for all K critical transistors in the face of variation. Using similar reasoning to the delay case, this means that 50% of our chips will need a voltage of $V_{50\%}$ or higher given by:

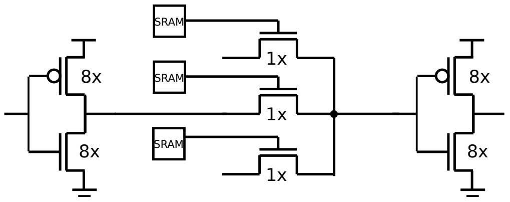

Figure 6: Directional switch circuit (3-input, 8× sized)

$$V_{50\%} = \mu_{V_{dd}} + N_\sigma(K) \times \sigma_{V_{th}} \qquad (19)$$

For example, if K=10^4, V_{th}=300mV, $\mu_{V_{dd}}$=700mV, and $\sigma_{V_{th}}$=20% of V_{th}, or 60mV, $N_\sigma\left(10^4\right)$=3.8, and V_{dd} must be 928mV or higher 50% of the time. *The more critical transistors in a design, the higher we must expect to set the supply voltage to maintain target performance.*

4. METHODOLOGY

To evaluate the delay and energy benefits of component-specific routing, we use the predictive technology models (PTM) [36] to model delay and energy of gates as a function of variation, and employ a modified version of VPR 5.0.2 [22] to perform component-specific routing and measure total chip delay and energy.

4.1 Variation, Delay & Energy Model

We calculate individual switch delays and energies by performing extensive HSPICE simulations using the high performance PTM. We simulate at several values of V_{dd}, V_{thp} and V_{thn} and curve fit to create a continuous model of device delay and energy as a function of V_{dd} and V_{th}. This results in highly accurate models (within 1% error compared to HSPICE across $0.1V < V_{dd} < 1.2V$ and 6 $\sigma_{V_{th}}$).

To model variation for routed designs, we modify VPR such that every switch uses a randomly generated set of V_{th}'s sampled from a Gaussian distribution. Coupled with V_{dd}, we use the curve-fit models from HSPICE to compute the delay and energy of individual circuit elements. We use this method to generate 50 chips that we then keep constant across our comparisons.

To model delay and energy of a full routed design, we make additional modifications to VPR to compute both dynamic and static energy; we calculate dynamic energy by summing up all switched capacitance, and we compute static energy by summing up the leakage power of all devices. To calculate switching activity (α in Eq. 4) for dynamic energy we use the ACE 2.0 switching activity estimator [17] with random (50%) input probabilities. While we modify VPR to measure energy, we do not change VPR's cost function to target energy minimization.

An important note about our variation and energy model is that, as a simplification, we only model random WID V_{th} variation and energy dissipation in interconnect logic of FPGAs. Random variation is expected to dominate future sources of variation; interconnect switching energy is the dominant source of energy in FPGAs today [34] (e.g. Tuan shows 62% of dynamic energy in routing with the rest split evenly between logic and clocking). We assume that SRAM variation is controlled by dual V_{dd}/V_{th} processes as is standard in modern commercial FPGAs [14, 34].

4.2 Architecture Model

We route on an architecture with 6-input LUTs and with 8 LUTs, 8 output pins and 27 input pins per CLB. We add additional CLB input and output pins to be utilized by delay-aware routing for additional routing flexibility (Sec. 3.3 and 5.1). All results are presented using 20% more channels than the minimum number of channels C_{min} required to route the particular benchmark design (Table 1).

The directional switch circuit we model is shown in Fig. 6. We use segment length 8, Wilton style S-Boxes and a C-Box connectivity of F_{cin}=F_{cout}=0.25. When considering different gate sizes, we uniformly size up the input and output inverters of the circuit. For simplicity, we only present results for single-stage buffers. The impact of variation on multi-stage buffers in terms of slew rate effects, failure and speed tradeoffs, and energy efficiency is sufficiently complex to warrant future work. By examining single-stage buffers we build a baseline on which to understand more complex buffering schemes.

4.3 VPR Noise Reduction

VPR 5.0.2 is known to introduce experimental noise by producing inconsistent results in routing. This effect is magnified when mapping to resources that each have different delays; moreover, with high variation and low V_{dd} these delays can vary by several orders of magnitude. To minimize router noise we used the timing-targeted router [28]. We route using 200 iterations and a `-max_crit` value of 0.9999.

4.4 Experimental Setup

We compare delay-aware routing to delay-oblivious routing under variation for the Toronto 20 benchmark set [4]. We perform clustering and placement in VPR and use a single placement per benchmark for all routing experiments. Each data point is obtained by running both routers on a set of 50 Monte Carlo generated chips with V_{th} variation. The 50 chips are routed individually by the delay-aware router, while the delay-oblivious router performs a single, nominal route and evaluates that route across all chips. We report all delay and energy data at the 90% parametric yield point (i.e. we discard the 5 slowest/highest energy chips and report the max delay and energy). With 50 Bernoulli trials the 90% confidence interval for the results reported as 90% yield is 85–95%.

5. RESULTS

To quantify the delay and energy benefits of component-specific mapping for FPGAs under variation, we compare delay-aware and delay-oblivious routers through a series of experiments. First, we examine how extra resources can improve functional yield, which is necessary to operate under high V_{th} variation and low voltage. Because sizing of transistors is critical to delay, energy, and variation tolerance, we also compare both routers across a variety of switch sizes. Then we choose delay/energy optimal sizing for both the delay-aware and delay-oblivious routers to make an accurate comparison. Finally, we show how these trends scale across technologies.

5.1 CLB IO Sparing

Sec. 2.1 detailed how parameter variation can lead to functional logic failures, particularly at very low voltages. In order for delay-aware routing to avoid these defects, there must be enough spare resources (Sec. 3.3) for the router to find an alternative, non-defective path for a given net. FPGAs naturally have many spare resources in the form of multiple IO pins per CLB, extra channels, and flexibility in C-Box and S-Box connectivity. However, in examining the ability of delay-aware routing to avoid defects, we found that CLB IO pins are a significant bottleneck for defect avoidance. The single buffer that brings an input into the CLB, or the single buffer that fans out to C-Box switches, serve as single points of failure that can render large amounts of connectivity unusable. If a CLB is highly populated, defective IO pins can make it impossible to route.

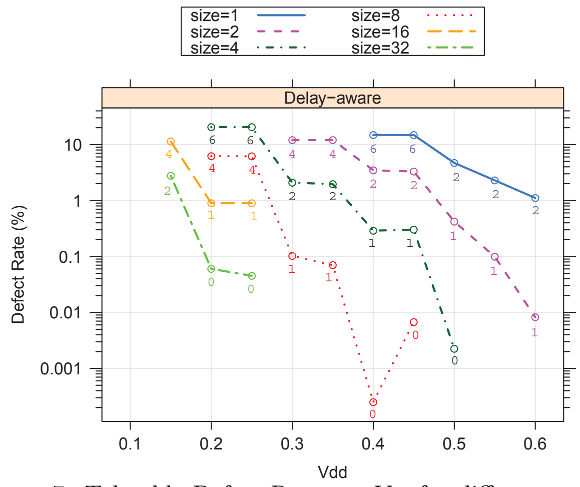

Figure 7: Tolerable Defect Rates vs V_{dd} for different extra IO pin counts (`des`, 22nm), all switch sizes

One approach to mitigate this problem is to add spare pins to be used exclusively for defect avoidance. In Fig. 7 we examine the ability of the delay-aware router to yield by plotting tolerable defect rate as a function of voltage at 22nm for the `des` benchmark. As defect rate is highly dependent upon the magnitude of variation, and this magnitude is dictated by sizing (Eq. 8), we also examine routability as a function of switch size. Each point in the graph is numbered to indicate the number of extra pins required to yield at the given voltage and size. At high enough voltages (for example, above 500mV for size 4), we observe no defective switches. As we lower voltage we see defects appear and increase sharply; as expected we see higher defects rates at smaller sizes for a fixed V_{dd} due to the increased variation. We see that increasing numbers of spare CLB IO pins are required to yield as voltage is lowered. However, adding extra pins increases area, energy, and delay as accounted for in our models. For simplicity, in the remaining experiments in this paper we select 2 spare CLB pins as a configuration which generally provides defect avoidance for rates $< 5\%$ with minimal cost ($< 1\%$ energy/operation).

5.2 Delay

Fig. 8 plots parametric delay as a function of V_{dd} across a series of switch sizes, for nominal, delay-oblivious, and delay-aware routes for `des` at 22nm. For the nominal, no variation case, we see that, at higher voltages, size 8 switches generally provide a good tradeoff between drive strength and capacitive load, which corroborates prior work in determining delay optimal switch sizes [18]. As we reduce V_{dd}, the

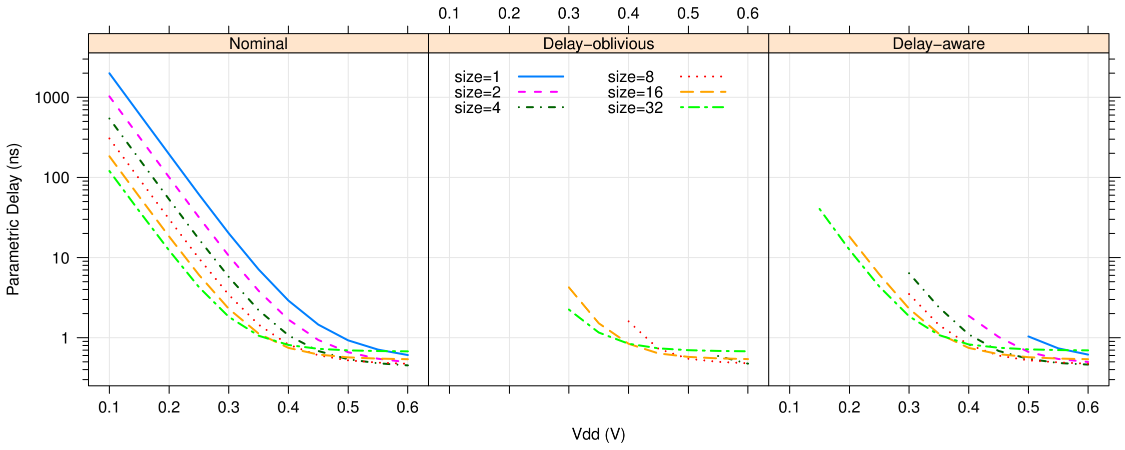

Figure 8: Parametric delay vs V_{dd} (des, 22nm), 2 extra pins, all switch sizes

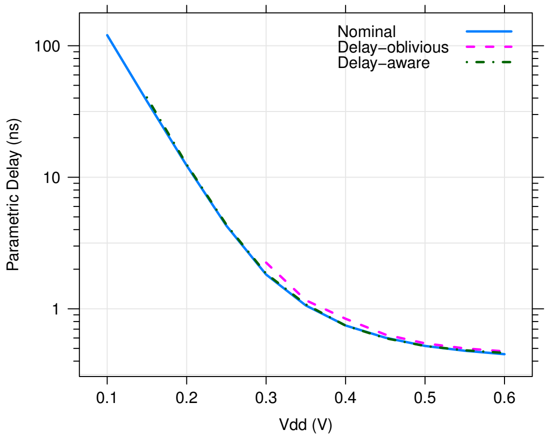

Figure 9: Parametric delay vs V_{dd} (des, 22nm), 2 extra pins, delay optimal switch sizes

increased drive strength of larger switches is necessary to reduce delay; below 400mV it is desirable to size up to 32× to minimize delay. The same basic delay-optimal sizing trends hold for delay-oblivious and delay-aware routing.

As we drop V_{dd} in the delay-oblivious case, we begin to see functional failures as described in Sec. 2.1 and shown in Fig. 7. The delay curves end at voltages where the defect rate becomes too high to achieve 90% yield. At 22nm there is sufficient variation ($\sigma_{V_{th}}/\mu_{V_{th}} = 27\%$) that enough small static CMOS inverters at low V_{dd} fail to switch which delay-oblivious cannot avoid. As we increase switch size and hence decrease the magnitude of variation, we are able to scale down V_{dd} and remain operational.

We see similar effects for the delay-aware router, where functional failures occur for small switches at low voltages. However, the delay-aware router is able to remain functional for lower voltages and smaller switch sizes through defect avoidance. For example, the delay-aware router can retain 90% yield for 32× sized gates at a V_{dd} that is 150mV lower than the delay-oblivious case (300mV vs 150mV).

Fig. 9 plots parametric delay for delay-optimal sizes across all V_{dd}'s (i.e. the composite minimum curve of Fig. 8). For delay-optimized sizing, here we can effectively see the delay margins induced by variation as a function of V_{dd} by comparing the nominal and delay-oblivious curves. At high V_{dd} these margins are typically negligible, less than 2%. However, as we drop the supply voltage these margins increase, up to around 1.2× the nominal delay at 300mV. Delay-aware routing is able to completely eliminate variation induced delay margins, and improve delay with respect to delay-oblivious routing by 1.2×.

5.3 Energy

Fig. 10 plots parametric energy as a function of V_{dd} and switch sizes. In the no variation case we observe energy beginning to minimize around 150mV, similar to that in Fig. 3. At low voltages, delay is increased, and therefore static energy/operation increases as we spend more time leaking in a single operation. We also see that minimum sized gates always provide energy-minimal operation, a well known result in subthreshold circuit design [1, 12].

The delay-oblivious graph shows the same functional yield issues as in Fig. 8: as V_{dd} is reduced, the delay-oblivious router fails to provide 90% functional yield. In order to achieve reduced energy at lower V_{dd}'s, we must increase gate sizes in order to avoid defects. We see that 16× sized switches provide the minimal energy per operation.

For the delay-aware case we also see functional failures; however, delay-aware routing extends the range over which gates can function through defect avoidance. For example, in the 16× case, delay-aware routing enables a voltage reduction compared to delay-oblivious routing of 100mV (from 300mV to 200mV).

Fig. 11 plots parametric energy for energy optimal sizes across all V_{dd}'s (again, the composite minimum curves of Fig. 10). When comparing delay-oblivious to nominal, we see that the energy margins induced by variation are substantial, around a factor of 2 in the worst case at 300mV. These margins are significant because, while the nominal case can optimally use minimum sized devices, the delay-oblivious case must increase gate size to continue to yield, which increases switched capacitance. For example, 16× sized gates are 16 times more capacitive, but because gate capacitance contributes ≈10% of total switched capacitance in the size=1 case (wire capacitance accounts for the remain-

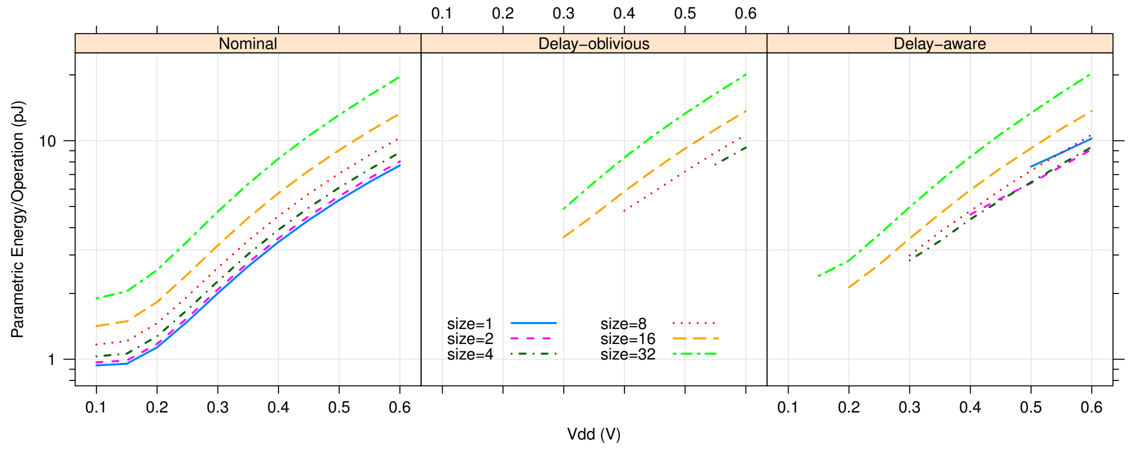

Figure 10: Parametric energy/operation vs V_{dd} (des, 22nm), 2 extra pins, all switch sizes

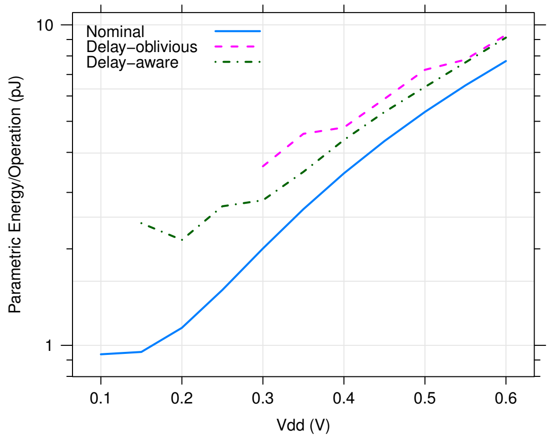

Figure 11: Parametric energy/operation vs V_{dd} (des, 22nm), 2 extra pins, energy optimal switch sizes

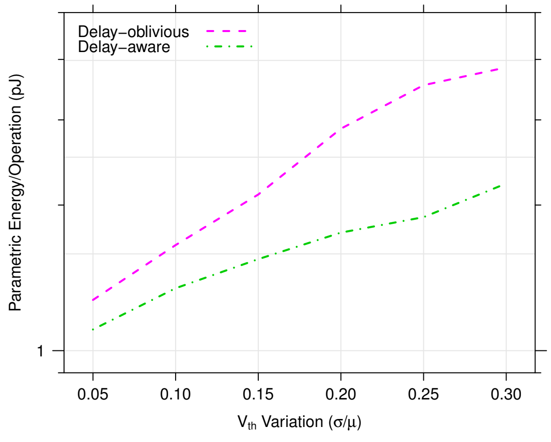

Figure 13: Minimum energy/operation vs V_{th} Variation (des, 22nm) 2 extra pins, energy optimal switch sizes

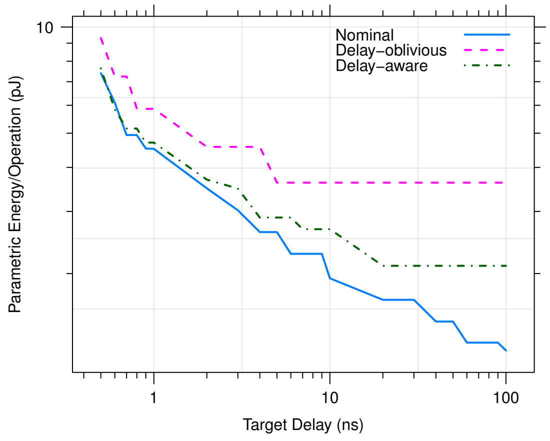

Figure 12: Parametric energy/operation vs delay target (des, 22nm), 2 extra pins, energy optimal switch sizes

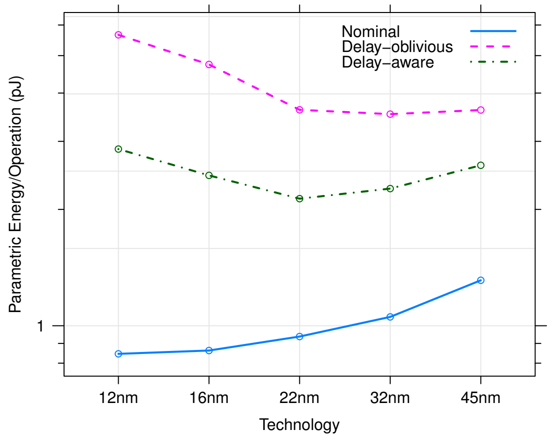

Figure 14: Minimum energy/operation vs technology (des) 2 extra pins, energy optimal switch sizes

ing 90%), increasing gate capacitance by an order of magnitude only increases switched capacitance 2×. We see that delay-aware routing is able to cut the energy margin nearly in half at 300mV, but is not able to complete eliminate it. Delay-aware routing is also able to prolong the range over which we can scale down V_{dd} and still reduce energy from 300mV to 200mV.

Table 1 shows the energy benefits of component-specific mapping as a function of technology (delay-oblivious/delay-aware). It also shows the average energy margins induced by variation across all benchmarks (delay-oblivious/nominal). On average across all benchmarks, variation increases minimum energy/operation by 2.12–6.12×, and delay-aware routing is able to improve minimum energy/operation by 1.42–1.98× as technology scales.

Fig. 12 demonstrates the energy benefits of delay-aware routing when targeting minimal energy operation under a performance constraint (as opposed to minimal energy ignoring delay). At larger delay targets, delay-aware routing achieves the required cycle times at a lower V_{dd} and hence lower energy/operation than delay-oblivious routing, with an energy savings of 1.2× for the fastest target of 2GHz and 1.8× for the slowest target of 10MHz.

5.4 Sensitivity to V_{th} Variation

The ITRS predictions for V_{th} variation are generally considered to be pessimistically high. This is exhibited by our very high defect rates at low voltages. To examine the benefits of delay-aware routing at various values of $\sigma_{V_{th}}$, Fig. 13 plots minimum energy/operation as a function of variation for des at 22nm, where the ITRS nominally predicts $\sigma_{V_{th}}/\mu_{V_{th}} = 27\%$. At 5% variation the ratio of delay-oblivious to delay-aware energy is only 1.15×. As variation increases to 30% we see the ratio scale up to 1.74×.

5.5 Technology Scaling

To analyze how our results scale with respect to feature size, we plot minimum energy/operation as a function of technology in Fig. 14. As technology scales, V_{th} variation increases substantially (we use ITRS predicted values of V_{th} as shown in Fig. 1), and we expect that delay-oblivious routing will have increased energy margins, while delay-aware routing may be able to reduce these margins. When examining the no variation case, we see energy/operation decrease with each technology generation, as expected. We also see substantial energy margins in the delay-oblivious case that delay-aware routing is able to reduce. However, we also note an interesting trend: the delay-oblivious case shows a slight net *increase* in energy beginning at the 32nm generation, due to the overhead of variation. This corresponds directly to the result from [7] that demonstrated a similar trends for ASICs, and a similar turning point at 32nm. When examining delay-aware routing, we see a similar increase, but below the 22nm generation. *This means that delay-aware routing is effectively able to allow technology scaling to continue delivering reductions in minimum operating energy for another technology generation.*

6. FUTURE WORK

This work has only explored part of the design space in evaluating component-specific routing to improve delay and energy. Perhaps most importantly, we have only examined single-stage buffers for simplicity. Multi-stage buffers are more realistic and representative switch circuits, but they will exhibit complex, composite effects from the results shown here (e.g. small buffers will dominate functional yield trends, but large buffers will contribute most to delay trends). Adding models for LUT variation, short-circuit power, and glitch power will also make our physical model more complete; nonetheless, we expect that their additions will not change our results significantly since interconnect delay and switching/leakage energy are dominant.

Additionally, we are limited by our selection of benchmarks in the Toronto 20. These circuits are noticeably small (see Table 1) and half are completely combinational; modern circuits are much larger and pipelined more aggressively. From the analysis in Sec. 3.1, 3.2 and 3.4, we expect delay-aware routing to show larger benefits for pipelined circuits and delay-oblivious routing to suffer greatly from the large number of near critical paths. Additionally, combinational circuits leak substantially more per operation due to the increased length of operation; pipelined circuits will use less energy per operation at low voltages, reducing the minimum energy point (Fig. 3).

Additional circuit techniques may further help to improve the energy benefits of delay-aware routing. At the minimum energy operating points, two-thirds of the energy is in leakage, and most of that leakage comes from unused devices. Power gating significantly reduces leakage energy/operation by disabling the many unused, leaky devices on a chip; delay-aware routing can further help by identifying the most leaky switches for gating. Selective sizing of gates may also help to reduce energy; instead of sizing up all gates to avoid defects, we can strategically size up gates that are most critical (such as CLB IO pins).

Table 1: Ratio of minimum energy per operation of delay-oblivious/delay-aware (90% yield, 2 extra IO pins)

Design	**LUT**	C_{min}	**Technology (nm)**				
			45	**32**	**22**	**16**	**12**
alu4	1492	52	1.69	1.81	1.87	2.02	2.02
apex2	1876	76	1.56	1.73	1.82	1.96	2.04
apex4	1304	78	1.29	1.46	1.57	1.87	1.90
bigkey	1816	72	1.49	1.72	1.88	2.58	2.69
clma	7808	92	1.42	1.61	1.73	2.06	2.09
des	1504	78	1.39	1.55	1.27	1.93	1.97
diffeq	1280	76	1.30	1.41	1.65	2.09	2.12
dsip	1372	64	1.58	1.82	1.90	2.01	2.05
elliptic	2784	60	1.88	2.08	2.19	2.35	2.45
ex1010	4744	114	1.17	1.27	1.48	1.88	1.91
ex5p	1092	70	1.37	1.53	1.66	1.87	1.96
frisc	2892	82	1.74	2.00	2.09	2.21	2.25
misex3	1388	68	1.63	1.78	1.81	1.78	1.76
pdc	4616	96	1.46	1.60	1.63	1.60	1.59
s298	2020	60	1.31	1.47	1.62	1.84	1.79
s38417	6232	50	1.29	1.46	1.63	2.09	2.09
s38584.1	6064	60	1.35	1.61	1.68	1.91	1.90
seq	1724	78	1.62	1.74	1.85	1.98	2.00
spla	3784	86	1.04	1.18	1.30	1.47	1.43
tseng	972	48	1.39	1.57	1.73	2.28	2.48
Geomean (benefit)			1.42	1.60	1.69	1.98	1.98
Geomean (margins[1])			2.12	3.04	4.01	5.04	6.12

[1] margin = delay-oblivious/nominal

7. CONCLUSIONS

We show that circuits mapped using component-specific routing and delay knowledge can mitigate delay and energy induced margins from variation—**knowledge is power** [2] (or more precisely, *energy*). For a standard set of FPGA benchmark circuits mapped to 45nm, 32nm, 22nm, 16nm and 12nm predictive technologies we show that, routing oblivious to variation yields an average energy overhead of 2.12–6.12×. Routing with post-fabrication delay knowledge can eliminate delay margins, and on average, reduces minimum energy/operation relative to delay-oblivious design by 1.42–1.98×. We further show that delay-aware routing can help extend minimum energy technology scaling by an extra generation. We hope this will motivate future work in solving the significant challenges in component-specific mapping.

Acknowledgments

This research was funded in part by National Science Foundation grant CCF-0904577. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

[1] International technology roadmap for semiconductors. <`http://www.itrs.net/Links/2010ITRS/Home2010.htm`> , 2010.

[2] F. Bacon. *Meditationes Sacræ. De Hæresibus*. 1597.

[3] K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer. High-performance CMOS variability in the 65-nm regime and beyond. *IBM J. Res. and Dev.*, 50(4/5):433–449, July/September 2006.

[4] V. Betz and J. Rose. FPGA Place-and-Route Challenge. <`http://www.eecg.toronto.edu/~vaughn/challenge/challenge.html`> , 1999.

[5] S. Bijansky and A. Aziz. TuneFPGA: post-silicon tuning of dual-Vdd FPGAs. In *DAC*, 2008.

[6] D. Bol, R. Ambroise, D. Flandre, and J.-D. Legat. Interests and limitations of technology scaling for subthreshold logic. *IEEE Trans. VLSI Syst.*, 17(10):1508–1519, 2009.

[7] D. Bol, R. F. Ambroise, and J.-D. D. Legat. Impact of technology scaling on digital subthreshold circuits. In *ISVLSI*, pages 179–184, 2008.

[8] L. Cheng, J. Xiong, L. He, and M. Hutton. FPGA performance optimization via chipwise placement considering process variations. In *FPL*, pages 1–6, 2006.

[9] C. Chow, L. Tsui, P. Leong, W. Luk, and S. Wilton. Dynamic voltage scaling for commercial FPGAs. *ICFPT*, pages 173–180, Dec. 2005.

[10] W. B. Culbertson, R. Amerson, R. Carter, P. Kuekes, and G. Snider. Defect tolerance on the TERAMAC custom computer. In *FCCM*, pages 116–123, April 1997.

[11] B. Gojman and A. DeHon. VMATCH: Using Logical Variation to Counteract Physical Variation in Bottom-Up, Nanoscale Systems. In *ICFPT*, pages 78–87. IEEE, December 2009.

[12] S. Hanson, B. Zhai, K. Bernstein, D. Blaauw, A. Bryant, L. Chang, K. K. Das, W. Haensch, E. J. Nowak, and D. M. Sylvester. Ultralow-voltage, minimum-energy CMOS. *IBM J. Res. and Dev.*, 50(4–5):469–490, July/September 2006.

[13] K. Katsuki, M. Kotani, K. Kobayashi, and H. Onodera. A yield and speed enhancement scheme under within-die variations on 90nm LUT array. In *CICC*, pages 601–604, 2005.

[14] M. Klein. The Virtex-4 power play. *Xcell Journal*, (52):16–19, Spring 2005.

[15] A. Kumar and M. Anis. FPGA Design for Timing Yield Under Process Variations. *IEEE Trans. VLSI Syst.*, 18(3):423–435, March 2010.

[16] J. Kwong and A. P. Chandrakasan. Variation-Driven device sizing for minimum energy sub-threshold circuits. In *ISLPED*, pages 8–13, 2006.

[17] J. Lamoureux and S. Wilton. Activity estimation for field-programmable gate arrays. *FPL*, pages 1–8, Aug. 2006.

[18] G. Lemieux, E. Lee, M. Tom, and A. Yu. Directional and single-driver wires in fpga interconnect. In *ICFPT*, pages 41–48, December 2004.

[19] Y. Lin, L. He, and M. Hutton. Stochastic physical synthesis considering prerouting interconnect uncertainty and process variation for FPGAs. *IEEE Trans. VLSI Syst.*, 16(2):124, 2008.

[20] Y. Lin, M. Hutton, and L. He. Placement and timing for FPGAs considering variations. In *FPL*, 2006.

[21] G. Lucas, C. Dong, and D. Chen. Variation-aware placement for FPGAs with multi-cycle statistical timing analysis. In *FPGA*, pages 177–180, New York, New York, USA, 2010. ACM.

[22] J. Luu, I. Kuon, P. Jamieson, T. Campbell, A. Ye, W. M. Fang, and J. Rose. VPR 5.0: Fpga cad and architecture exploration tools with single-driver routing, heterogeneity and process scaling. In *FPGA*, pages 133–142, 2009.

[23] M. Majzoobi, E. Dyer, A. Elnably, and F. Koushanfar. Rapid FPGA delay characterization using clock synthesis and sparse sampling. In *Proc. Intl. Test Conf.*, 2010.

[24] Y. Matsumoto, M. Hioki, T. Kawanami, T. Tsutsumi, T. Nakagawa, T. Sekigawa, and H. Koike. Performance and yield enhancement of FPGAs with within-die variation using multiple configurations. In *FPGA*, pages 169–177, 2007.

[25] G. Nabaaz, N. Aziziy, and F. N. Najm. An adaptive FPGA architecture with process variation compensation and reduced leakage. In *DAC*, pages 624–629, 2006.

[26] S. R. Nassif, N. Mehta, and Y. Cao. A resilience roadmap. In *DATE*, March 2010.

[27] R. Rubin and A. DeHon. Choose-Your-Own-Adventure Routing: Lightweight Load-Time Defect Avoidance. In *FPGA*, pages 23–32, 2009.

[28] R. Rubin and A. DeHon. Timing-Driven Pathfinder Pathology and Remediation: Quantifying and Reducing Delay Noise in VPR-Pathfinder. In *FPGA*, pages 173–176, 2011.

[29] P. Sedcole and P. Y. K. Cheung. Within-die delay variability in 90nm FPGAs and beyond. In *ICFPT*, pages 97–104, 2006.

[30] P. Sedcole and P. Y. K. Cheung. Parametric yield in FPGAs due to within-die delay variations: A quantitative analysis. In *FPGA*, pages 178–187, 2007.

[31] S. Sivaswamy and K. Bazargan. Variation-aware routing for FPGAs. In *FPGA*, pages 71–79, 2007.

[32] S. M. Trimberger. Utilizing multiple test bitstreams to avoid localized defects in partially defective programmable integrated circuits. United States Patent Number: 7,424,655, September 9 2008.

[33] T. Tuan, A. Lesea, C. Kingsley, and S. Trimberger. Analysis of within-die process variation in 65nm FPGAs. In *ISQED*, pages 1 –5, March 2011.

[34] T. Tuan, A. Rahman, S. Das, S. Trimberger, and S. Kao. A 90-nm Low-Power FPGA for Battery-Powered applications. *IEEE Trans. Computer-Aided Design*, 26(2):296–300, 2007.

[35] J. S. J. Wong, P. Sedcole, and P. Y. K. Cheung. Self-measurement of combinatorial circuit delays in FPGAs. *ACM Tr. Reconfig. Tech. and Sys.*, 2(2):1–22, 2009.

[36] W. Zhao and Y. Cao. New generation of predictive technology model for sub-45 nm early design exploration. *IEEE Trans. Electron Dev.*, 53(11):2816–2823, 2006.