Device And Architecture Co-Optimization for FPGA Power Reduction

ABSTRACT

Device optimization considering supply voltage Vdd and threshold voltage Vt tuning does not increase chip area but has a great impact on power and performance in the nanometer technology. This paper studies the simultaneous evaluation of device and architecture optimization for FPGA. We first develop an efficient yet accurate timing and power evaluation method, called trace-based model. By collecting trace information from cycle-accurate simulation of placed and routed FPGA benchmark circuits and re-using the trace for different Vdd and Vt, we enable the device and architecture co-optimization for hundreds of combinations. Compared to the baseline FPGA which has the architecture same as the commercial FPGA used by Xilinx, and has Vdd suggested by ITRS but Vt optimized by our device optimization, architecture and device co-optimization can reduce energy-delay product by 53.7% without any chip area increase compared to the conventional FPGA architecture. Furthermore, considering power-gating of unused logic blocks and interconnect switches, our co-optimization method reduces energy-delay product by 77% with a 30% chip area increase due to sleep transistors for power gating. To the best of our knowledge, this is the first in-depth study on architecture and device co-optimization for FPGAs.

1. INTRODUCTION

Field programmable gate array (FPGA) allows the same silicon implementation to be programmed or re-programmed for a variety of applications. It provides low NRE (non-recurring engineering) cost and short time to market. FPGA architecture has a significant impact on its performance, area, and power. Earlier architecture evaluation has been conducted to study the performance and area impacts of lookup table (LUT) size K (number of inputs of an LUT) and cluster size N (number of LUTs per cluster) [1, 2, 3]. As technology continues scaling down to the nanometer feature size, (e.g., 100nm or below), power has become an important design constraint for FPGAs. Recent studies [4, 5] developed parameterized FPGA power models and evaluated power characteristics of existing FPGA architectures.

To reduce FPGA power, several circuits and architectures have been proposed, including region based power gating of unused FPGA logic blocks [6], dual-Vdd and field programmability of Vdd for FPGA logic [7, 8] and interconnect [9]. However, no FPGA architecture evaluation has been published considering the above low-power FPGA circuits and architectures. In addition, the supply voltage (Vdd) and threshold voltage (Vt) have great impact on power (especially leakage power) and delay in nanometer technologies. However, all the aforementioned architecture evaluation assumed fixed Vdd and Vt [1, 2, 3, 4, 5], and have not conducted simultaneous evaluation on device optimization such as Vdd and Vt tuning and architecture optimization on LUT and cluster size.

Vdd and Vt optimization has little or no area overhead compared to power gating and Vdd programmability. Architecture and device co-optimization is obviously able to give better power and performance tradeoff compared to architecture tuning alone. We define *hype-architecture* (in short, hype-arch) as the combination of device parameters and architectural parameters. The co-optimization requires the exploration of the following dimensions: cluster size N, LUT size K, supply voltage Vdd, and threshold voltage Vt. The total hype-arch combinations can be easily over a few hundreds and calls for accurate yet extremely efficient timing and power evaluation methods.

The existing FPGA power evaluation methods are based on cycle-accurate simulation [4] or logic transition density estimation [5]. Timing and power are calculated for each circuit element. Therefore, it is very time-consuming to explore the huge hype-arch solution space using methods from [4, 5]. The first contribution of this work is that we develop a trace-based estimator for FPGA power, delay, and area. We perform benchmark profiling and collect statistical information on switching activity, short circuit power ratio, critical path structure, and circuit element utilization rate for a given set of benchmark circuits (MCNC benchmark set in this paper). We then derive formulae that use the statistical information and obtain FPGA performance and power for a given set of architectural and device parameter values. Our trace-based estimator has a high fidelity compared to the cycle-accurate simulation [4] and has an average estimation error of 9.1% for power and of 3.1% for delay. We will show that our trace information depends only on FPGA architecture but is *insensitive* to device parameters. Therefore, once the trace information is collected for the benchmark set, the remaining runtime is negligible as the trace-based hype-arch evaluation is based on formulae and lookup tables. The trace collecting has the same runtime as evaluating FPGA architecture for a given Vdd and Vt combination using cycle accurate simulation [4]. It took one week to collect the trace for the MCNC benchmark set using eight 1.2GHz Intel Xeon servers. But all the hype-arch evaluation reported in this paper with over hundreds of Vdd and Vt combinations took a few minutes on one server.

The second contribution is that we perform the architecture and device co-optimization for a variety of FPGA classes. We explore different Vdd and Vt combinations in addition to the cluster size and LUT size combinations. For comparison, we obtain the baseline FPGA which uses the same architecture as the commercial FPGA used by Xilinx, and Vdd suggested by ITRS[10] but Vt optimized by our device optimization, which is significantly better than the one with no device optimization. Compared to the baseline FPGA,

architecture and device co-optimization can reduce energy-delay product (product of energy per clock cycle and critical path delay, in short, ED) by 53.7% without additional area. Furthermore, considering power-efficient FPGA architecture with power-gating capability for logic blocks and interconnect switches, our architecture and device co-optimization method reduces energy-delay product by 77% with 30% area increase due to sleep transistors for power gating. To the best of our knowledge, this is the first in-depth study on architecture and device co-optimization for FPGAs.

The rest of the paper is organized as follows. Section 2 presents our trace-based estimation models. Section 3 applies the new estimation method and performs the architecture and device co-optimization. Section 4 concludes this paper.

2. TRACE-BASED ESTIMATION

In this section, we first discuss the preliminaries of FPGA architecture and review the power model used in the cycle-accurate simulation [4]. We then present our trace-based estimation for FPGA power and delay, and discuss the general methodology for architecture evaluation.

2.1 Preliminaries

We assume the same cluster-based island style FPGA as previous work [3, 4]. A logic block is a cluster of fully connected Basic Logic Elements (BLEs), and the cluster size N is the number of BLEs in a logic block. Each BLE consists of one Lookup Table (LUT) and one flip-flop. For an island style routing structure, logic blocks are surrounded by programmable routing channels, and the routing wires in both horizontal and vertical channels are segmented by *routing switch blocks*. In this paper, we use a fixed routing architecture, i.e. fully buffered routing switches and uniform wire segment spanning 4 logic blocks, which is the best routing architecture for low power FPGA [7]; and we study the impact of the N and K on architecture optimization. ** Moreover, we assumed the route channel width (number of tracks in each routine channel) to be 1.2 times of the minimum route channel width (the minimum width to let the FPGA circuit be routable). ** Because there is a limited number of cluster size and LUT size combinations, the previous evaluation method based on cycle-accurate simulation can be applied when only architecture optimization is considered.

We define our baseline FPGA as the cluster-based island style FPGA architecture with a Vdd of 0.9v suggested by ITRS [10] at 70nm technology, LUT size of 4 and cluster size of 8 as the Xilinx FPGA, and a Vt of 0.3v, which is optimized by our Vt tuning for minimum ED product. If we use a Vt of 0.25V, the ED increases by 58%. This illustrates the benefit of Vt optimization and the quality of the baseline FPGA. Table 1 gives Vdd and Vt levels for the baseline FPGA and the evaluation ranges of Vdd, Vt, N and K.

Baseline FPGA device/arch parameter values			
Vdd	Vt	N	K
0.9v	0.3v	8	4
Value range for device/arch optimization			
Vdd	Vt	N	K
0.8v-1.1v	0.2v-0.4v	6-12	3-7

Table 1: Baseline hype-arch and evaluation ranges.

2.2 Review on Cycle-Accurate Simulation

Given the above FPGA architecture, a detailed power model has been proposed for cycle-accurate simulation [4]. It models switching power, short-circuit power, and leakage power. The first two types of power are called *dynamic power* and they can only occur when a signal transition happens. The switching power is due to the charging and discharging of load capacitance, and can be modeled as follows,

$$P_{sw} = 0.5f \cdot V_{dd}^2 \cdot \sum_{i=1}^{n} C_i S_i \quad (1)$$

where n is the total number of nodes, f is the clock frequency, Vdd is the supply voltage, C_i is the load capacitance for node i and S_i is the switching activity for node i. Short-circuit power occurs when there is a signal transition at a gate output and the pull-up and pull-down transistors conduct simultaneously for a short period of time. It is a function of signal transition time and load capacitance, and can be modeled as follows.

$$P_{sc} = P_{sw} \cdot \alpha_{sc}(t_r) \quad (2)$$

where t_r is the signal transition time and $\alpha_{sc}(t_r)$ is the ratio between short-circuit power and switching power. α_{sc} depends on transition time t_r. The third type of power, *leakage power*, is consumed when there is no signal transition for a gate or a circuit module. It is a function of technology, temperature, static input vector, and stack effect of the gate type. average leakage power of a circuit element at given temperature, Vdd and Vt can be characterized by doing SPICE simulation under different input vectors. In each clock cycle of simulation, the simulation under real delay model obtains the number of signal transitions as well as transition time of a circuit element and calculate its dynamic power. If the circuit element has no signal transition in that cycle, it only consumes leakage power. Also, leakage power is consumed by an active element too. Essentially, the cycle-accurate simulation is needed to get the switching activity as well as signal transition time under real delay model.

2.3 Trace-based Estimation

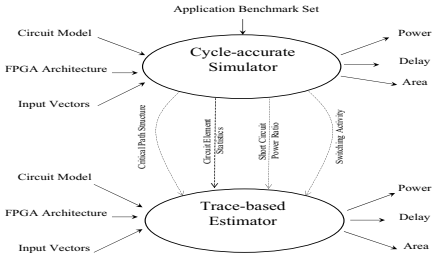

Figure 1: Cycle-accurate simulation versus trace-based estimation.

The cycle-accurate simulation is very time consuming because a large number of the input vectors needs to be simulated using detailed delay model. Also, in order to obtain FPGA delay, static timing analysis has to be conducted for the entire circuit mapped to the FPGA fabric. The cycle-accurate simulation is not practical for architecture and device co-optimization because the total hype-arch combinations can be easily over a few hundreds. We develop a runtime efficient trace-based estimation method. For a given benchmark set and a given FPGA architecture, we collect statistical information of switching activity, critical path structure and circuit element utilization by profiling the benchmark circuits using cycle-accurate simulation. These statistical information is called the *trace* of the given benchmark set. We further develop

Trace Parameters (depend on architecture)	
N_i^u	# of *used* circuit elements of resource type i
N_i^t	total # of circuit elements in resource type i
S_i^u	avg. switching activity for a *used* ckt element of type i
N_i^p	# of circuit elements of type i on the critical path
α_{sc}	ratio between short circuit power and switching power
Device Parameters (depend on technology)	
Vdd	power supply voltage
Vt	threshold voltage
Circuit Parameters (depend on circuit design and device)	
P_i^s	avg. leakage power for a circuit element in resource type i
C_i^u	avg. load capacitance of a circuit element of resource type i
D_i	avg. delay of a circuit element in resource type i

Table 2: Trace information, device and circuit parameters.

our quick estimation formula based on trace information and circuit models at different technologies. We will show that the trace information is insensitive to the device parameters such as Vdd and Vt, and it can be reused during our device optimization to avoid the time-consuming cycle-accurate simulation. Figure 1 illustrates the relation between the cycle-accurate simulation and trace-based estimation. Table 2 summarizes the trace information we collect as well as the device and circuit parameters. In the table, trace parameters, including N_i^u, N_i^t, S_i^u, N_i^p and α_{sc}, are what only depend on FPGA architecture, device parameters, including Vdd and Vt, are what depend on technology scale, and circuit parameters, including P_i^s, C_i^u and D_i, are what depend on circuit design and device. The details of our trace-based models are discussed in the following.

2.3.1 Dynamic Power Model

Dynamic power includes switching power and short-circuit power. A circuit implemented on a FPGA fabric cannot utilize all circuit elements in FPGA because of the programmability. Dynamic power is only consumed by the utilized FPGA resources. Our trace-based switching power model distinguishes different types of used FPGA resources and applies the following formula:

$$P_{sw} = \sum_i \frac{1}{2} N_i^u \cdot f \cdot V_{dd}^2 \cdot C_i^{sw} \tag{3}$$

The summation is over different types of circuit elements, i.e., LUTs, buffers, input pins and output pins. For circuit elements in FPGA resource type i, C_i^{sw} is the average switching capacitance and N_i^u is the number of used circuit elements, f is the operating frequency. In this paper, we assume the circuit works in its maximum frequency, i.e., the reciprocal of the critical path delay. The switching capacitance is further calculated as follows,

$$\begin{aligned} C_i^{sw} &= (\sum_{j \in El_i} C_{i,j}/N_i^u) \cdot S_i^u \\ &= C_i^u \cdot S_i^u \end{aligned} \tag{4}$$

For the type i circuit elements, C_i^u is the average load capacitance of a used circuit elements, which is average over $C_{i,j}$, the local load capacitance for used circuit element j, El_i is the set of used type i circuit elements, and S_i^u is the average switching activity of used type i circuit elements. We assume that the average switching activity of the circuit elements is determined by the circuit logic functionality and FPGA architecture. The device parameters of Vdd and Vt have a limited effect on switching activity. We verify this assumption in Table 3 by showing the average switching activity of five benchmarks at different Vdd and Vt levels.

The short circuit power is related to signal transition time, which is difficult to obtain without detailed simulation under real delay

bench	70nm Vdd=1.1 Vt=0.25		100nm Vdd=1.3 Vt=0.32	
mark	logic	interconnect	logic	interconnect
alu4	2.06	0.55	2.01	0.54
apex2	1.73	0.47	1.75	0.47
apex4	1.23	0.27	1.19	0.26
bigkey	1.75	0.56	1.96	0.59
clma	0.90	0.21	0.87	0.21

Table 3: Switching activity comparison for different technology scale, Vdd and Vt.

model. In our trace-based model, we model the short circuit power as:

$$P_{sc} = P_{sw} \cdot \alpha_{sc} \tag{5}$$

Where α_{sc} is the ratio between short circuit power and switching power. Such ratio value is a circuit parameter depending on FPGA circuit design and architecture. We assume α_{sc} does not depend on device and technology scale.

For a given FPGA architecture (i.e, N and K), we profile each MCNC benchmark circuit to get the average switching activity for each resource type in the FPGA. The trace parameters α_{sc}, N_i^u and C_i^u depend only on the FPGA architecture and application benchmark set.

2.3.2 Leakage Power Model

The leakage power is modeled as follows,

$$P_{static} = \sum_i N_i^t P_i^s \tag{6}$$

For resource type i, N_i^t is the total number of circuit elements, and P_i^s is the leakage power for a type i element. Notice that usually $N_i^t > N_i^u$ because the resource utilization rate is low in FPGAs. For a FPGA architecture with power-gating capability, an unused circuit element can be power-gated to save leakage power. In that case, the total leakage power is modeled by the following formula:

$$P_{static} = \sum_i N_i^u P_i + \alpha_{gating} \cdot \sum_i (N_i^t - N_i^u) P_i \tag{7}$$

where α_{gating} is the average leakage ratio between a power-gated circuit element and a circuit element in normal operation. SPICE simulation shows that sleep transistors can reduce leakage power by a factor of 300 and $\alpha_{gating} = 0.003$ is used in this paper.

2.3.3 Delay Model

To avoid the static timing analysis for the whole circuit implemented on a given FPGA fabric, we obtain the structure of the ten longest circuit paths including the critical path for each circuit. The path structure is the number of elements of different resource types i.e., LUT, wire segment and interconnect switch, on one circuit path. We assume that the new critical path due to different Vdd and Vt levels is among these ten longest paths found by our benchmark profiling. When Vdd and Vt change, we can calculate delay values for the ten longest paths under new Vdd and Vt levels, and choose the largest one as the new critical path delay. Therefore, the FPGA delay can be calculated as follows:

$$D = \sum_i N_i^p D_i \tag{8}$$

For resource type i, N_i^p is the number of circuit elements that the critical path goes through, and D_i is the average delay of such a circuit element. D_i is the circuit parameter depending on Vdd, Vt, process technology and FPGA architecture. To get the path statistical information N_i^p, we only need to place and route the circuit *once* for a given FPGA architecture.

2.3.4 *Accuracy Verification*

In this section we show some experimental results to verify the accuracy of our power model. In our experiment, we test both 70nm and 100nm technology. We assume Vdd=1.0 and Vt=0.2 for 70nm technology, and Vdd=1.3 and Vt=0.32 for 100nm technology. The cluster size is from 6 to 12 and LUT size is from 3 to 7. We test 20 benchmarks under each architecture. For every architecture, power and delay are computed as the geometry mean of twenty benchmarks. Figure 2 illustrates the comparison of power and delay between the cycle simulation and the trace-based estimation. Compared to cycle-accurate simulation the average power error of trace-based estimation is 1.5% and average delay error is 4.2%. From the figure, the trace-based estimation will give the same trend of power and delay as cycle simulation. Therefore, it has a high fidelity. Moreover the run time of the trace-based estimation is $2s$, while that of the cycle-accurate simulation is 120 hours.

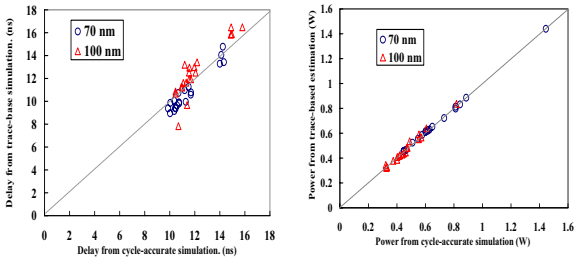

Figure 2: Comparison of power and delay between cycle-accurate simulation and trace-based estimation

3. HYPE-ARCH EVALUATION

3.1 Overview

In this section, we evaluate four FPGA hype-arch classes: $Class1$, $Class2$, $Class3$ and $Class4$. $Class1$ is the conventional FPGA using homogeneous-Vt for both interconnect and logic block (in short, homogeneous-Vt). $Class2$ applies different Vt to logic blocks and interconnects (in short heterogeneous-Vt). $Class3$ and $Class4$ are the same as $Class1$ and $Class2$, respectively, except that unused logic blocks and interconnects are power-gated. All these hype-arch classes are summarized in Table 4. We compare them with the baseline hype-arch, which has a cluster size of 8, LUT size of 4, Vdd of 0.9v (suggested by ITRS [10]), and Vt of 0.3v that is optimized with respect with the above architecture and Vdd. The base line hype-arch and evaluation ranges for device and architecture are shown in Table 1. Note that a high Vt is applied to all SRAM cells for configuration to reduce leakage power as suggested by [7].

In our study, we find that utilization rate of FPGA circuit (utilization rate is defined as the utilization rate of logic blocks, i.e., number of used logic blocks over total available logic blocks) does not affect the hype-arch evaluation. As shown in Table 5, the best hype-archs under different utilization rate are the same. Therefore throughout our following study we assume the logic block utilization rate to be 0.5. In Section 3.2, we study the effect of heterogeneous-Vt and power-gating, then in Section 3.4 we study the optimization of Vt with Vdd scaling.

3.2 heterogeneous-Vt and power-gating

Hype-arch Class	Case to study
Class1	homogeneous-Vt w/o power-gating
Class2	heterogeneous-Vt w/o power-gating
Class3	homogeneous-Vt w/ power-gating
Class4	heterogeneous-Vt w/ power-gating

Table 4: Summary of FPGA hype-arch Classes.

In this section, we present the hype-arch evaluation. For each hype-arch, we compute the energy and delay as the geometry mean of 20 MCNC benchmarks. If hype-arch A has less energy consumption and a smaller delay than hype-arch B, then we say that A is superior to B. We define the *dominant hype-arch* (in short, dom-arch) as the set of hype-archs that are not in superior to any other hype-archs.

Figure 3 presents the energy-performance trade-off for FPGA dom-archs of *Class 1* and *Class 2*. The min-ED hype-archs for all classes are summarized in Table 6. The optimal Vt for logic blocks (CVt) is higher than the Vt for interconnects (IVt) because the interconnect delay is more significant than logic block delay. Compared to the baseline hype-arch, *Class 1* reduces the min-ED by 40.1% and *Class 2* reduces the min-ED by 53.7%. By applying heterogeneous-Vt we can reduce ED without any area increase.

Hyper-arch. Class	Vdd (V)	CVt (V)	IVt (V)	(N, k)	ED (nJ· ns)	ED Reduction
Baseline	0.9	0.30	0.30	(8,4)	25.7	-
Class1	1.0	0.35	0.35	(6,6)	15.4	40.1%
Class2	1.0	0.40	0.30	(6,4)	11.9	53.7%
Class3	1.0	0.35	0.35	(8,5)	7.11	72.3%
Class4	1.0	0.35	0.30	(12,4)	5.76	77.6%

Table 6: Comparison between baseline and min-ED hype-arch in *Class 1*, *Class 2*, *Class 3*, and *Class 4*. Note: for the heterogeneous-Vt classes, i.e., Class1 and Class3, CVt=IVt.

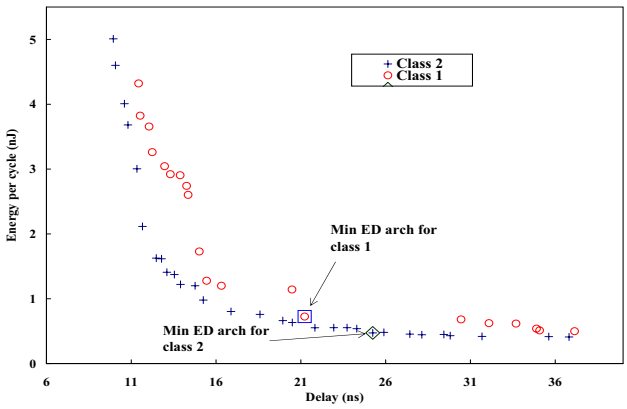

Figure 3: Dom-archs of Class 1 and Class 2.

Furthermore, power-gating can be applied to unused FPGA logic blocks and interconnect to reduce leakage power. Sleep transistors needed by power gating introduce delay and area overhead. The larger the sleep transistor, the smaller the delay overhead. Because only one sleep transistor is used for one logic block, we assume that a 210X PMOS transistor is used as the sleep transistor for a logic block and the area overhead is negligible. Because one sleep transistor is needed for an interconnect switch that is much smaller than a logic block, the area overhead can be significant. To

util rate	0.5					0.7					0.9				
	Vdd (V)	CVt (V)	IVt (V)	(N, k)	ED (nJ· ns)	Vdd (V)	CVt (V)	IVt (V)	(N, k)	ED (nJ· ns)	Vdd (V)	CVt (V)	IVt (V)	(N, k)	ED (nJ· ns)
Class1	1.0	0.35	0.35	(6,6)	15.4	1.0	0.35	0.35	(6,6)	11.3	1.0	0.35	0.35	(6,6)	8.91
Class2	1.0	0.40	0.30	(6,4)	11.9	1.0	0.40	0.30	(6,4)	8.93	1.0	0.40	0.30	(6,4)	6.87
Class3	1.0	0.35	0.35	(8,5)	7.11	1.0	0.35	0.25	(8,5)	6.47	1.0	0.35	0.25	(8,5)	6.11
Class4	1.0	0.35	0.30	(12,4)	5.76	1.0	0.35	0.30	(12,4)	5.31	1.0	0.35	0.30	(12,4)	5.05

Table 5: Min-ED hype-arch under different utilization rates.

determine the size of sleep transistors for interconnects, Figure 4 presents the chip-level delay-area trade-off for hype-arch{Vdd=1v, Vt=0.35v, (N,k)=(8,5) } with different sleep transistor sizes. In this figure, we find that 4X PMOS as the sleep transistor for a 7X routing buffer achieves the best delay and area trade-off. For the following study on power-gating, we use 4X sleep transistors for interconnects.

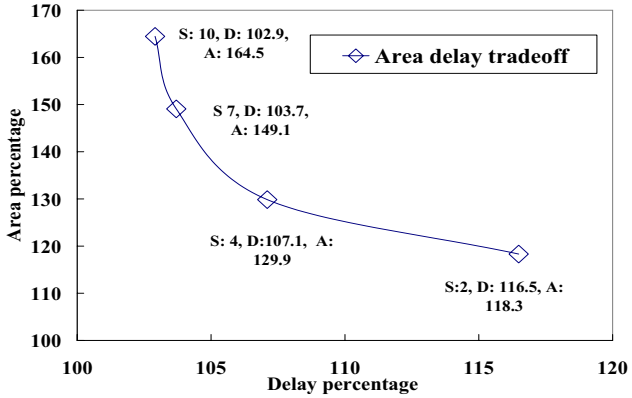

Figure 4: Delay and area trade-off with respect to different sleep transistor sizes in single Vt case. Note: for each node in the figure, S refers to the sleep transistor size. D and A refer to delay % and area % of circuit with power gating over that without power gating, respectively.

Figure 5 presents the energy-performance trade-off for FPGA dom-archs of *Class 3* and *Class 4*. Compared to the baseline hype-arch, *Class 3* reduces the min-ED by 72.3% and *Class 4* reduces min-ED by 77.6%, as shown in Table 6. Note that the power gap between Class3 and Class4 is smaller than that between Class1 and Class2 due to that the leakage power is significantly reduced by field programmable power gating and therefore the more detailed Vt tuning such as heterogeneous-Vt has a smaller impact.

Table 7 summarizes a few hyper-arches within a similar range of delay for the 4 classes. From the table, we observe the following: (1) heterogeneous-Vt decreases the LUT size in the min-ED hype-arch. For the min-ED hype-arch, IVt (Vt for interconnect) is lower than CVt (Vt for logic). This causes the interconnect power to increase. To compensate for the power increase, a smaller LUT size is used to reduce the logic power. (2) power-gating reduces the Vt of the min-ED hype-arch. When power-gating is applied, the leakage power of unused circuit elements is significantly reduced. Therefore a lower Vt can be used to improve performance but without much increase in leakage power.

3.3 Comparison of Device and Architecture Tuning

In this section, we compare the impact of device tuning and

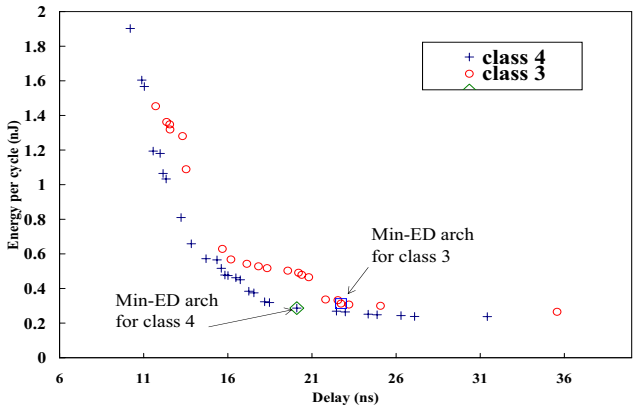

Figure 5: Dom-archs of *Class 3* and *Class 4*.

architecture tuning. In Figure 6, each set of data points is the dom-archs for a given device setting. For example, set D2 is the dom-archs under Vdd=1.0 and Vt=0.25. From the figure, we observe that a change on the device level leads to a more significant change in power and delay than architecture change does. For instance, for the set D7, where Vdd=0.9 and Vt=0.35, power for different architecture is between $0.285nJ$ and $0.311nJ$ and delay is between $31.0ns$ and $37.6ns$, however, if Vt becomes 0.30 as in set D6, power is between $0.450nJ$ and $0.509nJ$ and delay is between $19.6ns$ and $24.8ns$. As the impact of device tuning is much more significant than that of architecture tuning. Therefore, it is important for us to evaluate both device and architecture optimizations instead of evaluating architecture only.

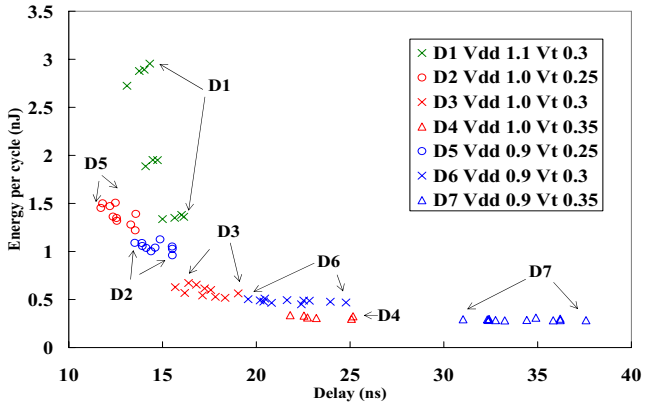

Figure 6: Dom-archs under different device setting.

3.4 Vt Optimization for chip-level Voltage Scaling

Class 1						Class 2						
Vdd (V)	Vt (V)	(N,k)	ED (nJ· ns)	E (nJ)	D (ns)	Vdd (V)	CVt (V)	IVt (V)	(N,k)	ED (nJ· ns)	E (nJ)	D (ns)
0.9	0.35	(8,5)	24.0	1.240	19.4	0.9	0.35	0.25	(6,4)	12.1	0.553	21.8
0.9	0.35	(6,5)	23.4	1.140	20.5	1.0	0.40	0.30	(8,5)	12.7	0.554	22.9
1.0	0.35	(6,6)	15.4	0.726	21.2	0.9	0.35	0.30	(6,5)	13.1	0.538	24.3
1.0	0.35	(8,6)	16.3	0.745	21.9	1.0	0.40	0.30	(6,4)	11.9	0.473	25.2
1.0	0.35	(10,5)	16.1	0.714	22.6	1.0	0.40	0.35	(6,6)	12.5	0.482	25.9
Class 3						Class 4						
Vdd (V)	Vt (V)	(N,k)	ED (nJ· ns)	E (nJ)	D (ns)	Vdd (V)	CVt (V)	IVt (V)	(N,k)	ED (nJ· ns)	E (nJ)	D (ns)
0.9	0.35	(6,6)	9.84	0.503	19.6	1.0	0.35	0.30	(8,5)	5.88	0.323	18.2
0.9	0.35	(10,5)	9.68	0.465	20.8	1.0	0.35	0.30	(10,5)	5.89	0.319	18.5
1.0	0.35	(8,6)	7.50	0.333	22.5	1.0	0.35	0.30	(12,4)	5.76	0.287	20.1
1.0	0.35	(8,5)	7.11	0.313	22.7	0.9	0.35	0.25	(6,4)	6.04	0.269	22.5
1.0	0.35	(10,5)	7.13	0.307	23.2	0.9	0.35	0.25	(12,4)	6.08	0.264	23.0

Table 7: Summary of 5 min-ED hype-archs in *Class 1*, *Class 2*, *Class 3*, and *Class 4*.

hyper-architecture Class	entire Vdd range $0.8v \sim 1.1v$			Vdd sub-range					ED reduced (%)
				0.8v, 0.9v		1.0v, 1.1v			
	Cvt (V)	Ivt (V)	weighted ED ($nJ \cdot ns$)	Cvt (V)	Ivt (V)	Cvt (V)	Ivt (V)	weighted ED ($nJ \cdot ns$)	
Class1	0.35	0.35	21.7	0.35	0.35	0.40	0.40	21.3	1.84%
Class2	0.40	0.35	18.4	0.35	0.25	0.40	0.35	15.9	13.6%
Class3	0.35	0.35	10.5	0.30	0.30	0.35	0.35	9.97	5.05%
Class4	0.35	0.30	9.45	0.35	0.25	0.40	0.35	8.25	12.7%

Table 8: Vt optimization with N=8, K=5.

Chip-level voltage scaling can be applied to FPGA to change Vdd level and reduce energy consumption without violating the timing specification for the current application in the just-in-time computation fashion. Given the distribution of Vdd levels for different application in an FPGA platform, the threshold voltage Vt of the chip can be optimized to minimize the weighted arithmetic mean of ED (weighted ED), where the weight is the given distribution.

Below, we assume the distribution of Vdd level 0.8v, 0.9v, 1.0v, and 1.1v to be 12.5%, 25%, 37.5%, and 25%, respectively. We find the optimal Vt for logic blocks and for interconnects to minimize the weighted mean-ED. Vt optimization is performed for the entire Vdd range from 0.8v to 1.1v, or two Vdd sub-ranges {0.8v, 0.9v} and {1.0v, 1.1v}. Instead of using uniform Vt over the entire Vdd range, using different Vt in two Vdd subranges may improve the weighted min-ED. However, using different Vt will increase design and fabrication cost because the producer may have to design two platforms for different Vdd subranges. Table 8 presents the weighted min-ED reduction by using two Vdd ranges over using one Vdd range for the four hype-arch classes. For hype-arch *Class1* and *Class3*, optimizing Vt in two Vdd subranges can only improve the weighted min-ED by 1.84% and 5.05%, respectively. However, for *Class2* and *Class4*, optimizing Vt in two Vdd subranges can improve the weighted ED by 13.6% and 12.7%, respectively. We can see that the heterogeneous-Vt hype-arch classes (*Class2* and *Class4*) have more weighted min-ED improvement than the homogeneous-Vt classes (*Class1* and *Class3*). It is because heterogeneous-Vt classes have more flexibility in Vt optimization. Optimizing Vt for different Vdd subrange therefore is worthwhile and can bring more improvement in heterogeneous-Vt hype-arch classes.

4. CONCLUSIONS AND DISCUSSIONS

In this paper, we have developed trace-based power and performance models for FPGA. The new models are much faster but yet accurate compared to the cycle-accurate simulation [4]. The one-time use of cycle-accurate simulation is applied to collect the timing and power trace for given benchmark set and given FPGA architecture. Then the trace can be re-used to calculate timing and power via closed-form formulae for different device parameters and technology scaling.

Using the trace-based estimation, we have first performed device (Vdd and Vt) and architecture (cluster and LUT size) co-optimizations for low power FPGAs. We assume the 70nm ITRS technology and use the following baseline for comparison: Vdd of 0.9v as suggested by ITRS, Vt of 0.3v as given by our Vt optimization for min-ED (i.e., minimum energy delay product), cluster size of 8 and LUT size of 4 as in Xilinx FPGA. Compared to the baseline case, simultaneous optimization of FPGA architecture, Vdd and Vt reduces the min-ED by 40% for FPGA using homogeneous-Vt for the logic and interconnect without power gating, and optimizing Vt separately (i.e., heterogeneous-Vt) for the logic and interconnect reduces min-ED by 53.7%. Furthermore, power gating unused logic and interconnect reduces the min-ED by up to 77% but increase the device area increases by 30% due to the sleep transistors needed by power gating. Compared to the homogeneous-Vt FPGAs, the min-ED hype-arch using heterogeneous-Vt has a smaller LUT size. In addition, device (i.e., Vdd and Vt) tuning has a more significant impact on power and delay than architecture tuning does.

Assuming FPGA chip-level Vdd scaling for just-in-time computation, we have then compared (i) one fixed and optimal Vt for the entire Vdd scaling range and (ii) two optimal Vt values for the two Vdd scaling subranges. Experiments show that finding optimal Vt for two subranges reduces the weighted min-ED by up to 5.05% for homogeneous-Vt FPGA and by up to 13.6% for hetergeneous-Vt FPGA, where the weight is the distribution of Vdd levels used for applications with different timing requirements.

5. REFERENCES

[1] J. Rose, R. J. Francis, D. Lewis, and P. Chow, "Architecture of field-programmable gate arrays: The effect of logic functionality on area efficiency," *Proc. IEEE Int. Solid-State Circuits Conf.*, 1990.

[2] J. Kouloheris and A. E. Gamal, "FPGA area vs. cell granularity - lookup tables and pla cells," in *1st ACM Workshop on FPGAs, berkeley, CA*, Feb 1992.

[3] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," in *Proc. ACM Intl. Symp. Field-Programmable Gate Arrays*, pp. 3–12, Feb 2000.

[4] F. Li, D. Chen, L. He, and J. Cong, "Architecture evaluation for power-efficient FPGAs," in *Proc. ACM Intl. Symp. Field-Programmable Gate Arrays*, Feb 2003.

[5] K. Poon, A. Yan, and S. Wilton, "A flexible power model for FPGAs," in *Proc. of 12th International conference on Field-Programmable Logic and Applications*, Sep 2002.

[6] A. Gayasen, Y. Tsai, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, and T. Tuan, "Reducing leakage energy in FPGAs using region-constrained placement," in *Proc. ACM Intl. Symp. Field-Programmable Gate Arrays*, February 2004.

[7] F. Li, Y. Lin, L. He, and J. Cong, "Low-power FPGA using pre-defined dual-vdd/dual-vt fabrics," in *Proc. ACM Intl. Symp. Field-Programmable Gate Arrays*, Februray 2004.

[8] F. Li, Y. Lin, and L. He, "FPGA power reduction using configurable dual-vdd," in *Proc. Design Automation Conf.*, June 2004.

[9] Fei Li, Yan Lin and Lei He, "Vdd programmability to reduce fpga interconnect power," in *Proc. Intl. Conf. Computer-Aided Design*, November 2004.

[10] International Technology Roadmap for Semiconductor in *http://public.itrs.net/*, 2002.