Circuits and Architectures for Field Programmable Gate Array with Configurable Supply Voltage

Yan Lin, Fei Li and Lei He

Electrical Engineering Department
University of California, Los Angeles

Abstract

Field Programmable Gate Arrays (FPGAs) with supply voltage (Vdd) programmability have been proposed recently to reduce FPGA power, where Vdd levels can be customized for FPGA circuit elements and unused circuit elements can be power-gated. In this paper, we first design novel Vdd-programmable and Vdd-gateable interconnect switches. Using the new switches, we propose three new classes of Vdd-programmable FPGA architectures. *Class1* applies Vdd programmability to both logic blocks and interconnects with configurable Vdd-level converters inserted before each interconnect switch. *Class2* uses Vdd-programmable logic blocks and Vdd-gateable interconnects. *Class3* is the same as *Class1* except that there is no Vdd-level converter in routing channels. Using a highly quantitative approach with placed and routed benchmark circuits, we then carry out architecture evaluation. Our baseline architecture class *Class0* uses high-Vdd for both logic blocks and interconnects, and is equivalent to the cutting-edge commercial products. High threshold voltage (Vt) is applied to configuration SRAM cells for all the four architecture classes, and the same fixed dual-Vdd levels are used for architecture Class1, Class2 and Class3. We define the energy-delay product (ED) as the geometric mean of energy-delay products over the benchmark set, and find the min-ED architecture, i.e.,the combination of cluster and LUT sizes that leads to a minimal ED for each architecture class. Compared to the min-ED architecture in FPGA Class0, the min-ED architectures in Class1, Class2 and Class3 reduce ED by 25.97%, 54.39% and 60.13%, respectively. The SRAM cell overhead due to Vdd programmability for Class1, Class2 and Class3 is 132%, 3% and 28%, respectively. The total device area overhead for Class1, Class2 and Class3 is 118%, 17% and 52%, respectively. Both FPGA Class2 and Class3 reduce more energy with less SRAM and area overhead compared to FPGA Class1. While FPGA Class3 gives the lowest energy consumption, FPGA Class2 achieves comparable energy reduction with significantly reduced SRAM and area overhead. Our evaluation results also show that LUT size 4 always gives the lowest energy consumption as well as the smallest total device area while LUT size 7 always leads to the highest performance.

Index Terms

Field programmable gate arrays, Architecture, Digital integrated circuits, Power supplies.

I. INTRODUCTION

FPGA provides an attractive design platform with low NRE (non-recurring engineering) cost and short time-to-market. Due to a large number of transistors for field programmability and the low utilization rate of FPGA resources, existing FPGAs consume more power compared to ASICs. Previous study [1] has shown that FPGA designs are highly power inefficient compared to their ASIC counterparts. As the process advances to nanometer technology and low-energy embedded applications are explored for FPGAs, power consumption becomes a crucial design constraint for FPGAs.

A few recent works have studied FPGA power modeling and optimization. [2], [3] present power evaluation frameworks for generic parameterized FPGA architectures and show that both interconnect and leakage power are significant for nanometer FPGAs. [4] analyzes the leakage power of a commercial FPGA architecture in 90nm technology and quantifies the leakage power challenge. FPGA power optimization involves CAD algorithms and novel circuits and architectures. [5] proposes configuration inversion method to reduce leakage power without any additional hardware cost. Leveraging the property of basic FPGA logic elements that are able to implement arbitrary functions with bounded input number, active leakage power is reduced by reconfiguring input vectors of multiplexers. [6] studies a suite of power-aware FPGA CAD algorithms without changing the existing FPGA circuits and architectures. It is shown that technology mapping and clustering algorithms are the most effective at reducing power, and the overlap between the energy savings by each CAD stage is small. The following work focuses on designing power-efficient FPGA circuits and architectures. [7] studies region-based power-gating to reduce leakage power of unused FPGA logic blocks. [8], [9] propose dual-Vdd and Vdd-programmable FPGA logic blocks to reduce both dynamic and leakage power. Novel dual-Vdd FPGA circuits and fabrics are presented and placement algorithms are developed to leverage the new fabrics.

Previously, conventional FPGA architecture evaluation has been performed using metrics of area, delay and energy in [2], [3], [10]. However, the emerging power-efficient circuits and architectures lead to different FPGA power characteristics, and therefore call for an architecture evaluation considering these power optimization techniques. In this paper, we study Vdd-programmable FPGAs which are originally proposed in [8], [9]. We first design a new set of Vdd-programmable circuits

This paper is partially supported by NSF CAREER award CCR-0093273, NSF grant CCR-0306682. We used computers donated by Intel. Address comments to lhe@ee.ucla.edu.

and develop several new architecture classes for Vdd-programmable FPGAs with Vdd programmability for interconnects. We then study the effect of cluster and LUT sizes on FPGA area, energy and delay, and evaluate the energy saving by our new Vdd-programmable architecture classes compared to FPGAs with a fixed Vdd level.

The rest of the paper is organized as follows. Section II first describes FPGA architecture background, evaluation methodology, power evaluation framework, and presents evaluation results for the baseline architecture class. Section III presents the novel circuit designs for Vdd-programmable and Vdd-gateable interconnect switches with minimal number of configuration SRAM cells. Sections IV and V propose three Vdd-programmable architecture classes and evaluate their energy, delay and area with comparison to the baseline case. We conclude this paper in Section VI.

II. Background

A. Cluster-based Island Style FPGAs

We assume cluster-based island style FPGA architecture such as that in [3], [11] for all classes of FPGAs studied in this paper. Figure 1 shows the cluster-based logic block, which includes N fully connected Basic Logic Elements (BLEs). Each BLE includes one k-input lookup table (LUT) and one flip-flop (DFF). The combination of cluster size N and LUT size k is the architectural issue we evaluate in this paper. The routing structure is of the island style shown in Figure 2. The logic blocks are surrounded by routing channels consisting of wire segments. The input and output pins of a logic block can be connected to the wire segments in routing channels via a *connection block* (see Figure 2 (b)). A routing *switch block* is located at the intersection of a horizontal channel and a vertical channel. Figure 2 (c) shows a subset switch block [12], where the incoming track can be connected to the outgoing tracks with the same track number[1]. The connections in a switch block (represented by the dashed lines in Figure 2 (c)) are programmable routing switches. We implement routing switches by tri-state buffers and use two tri-state buffers for each connection so that it can be programmed independently for either direction. We define an *interconnect segment* as a wire segment driven by a tri-state buffer or a buffer.[2] We use the smallest square array for each benchmark circuit, and decide the routing channel width CW in the same way as the architecture study in [11], [13], i.e., $CW = 1.2CW_{min}$ where CW_{min} is the minimum channel width required to route the given circuit successfully. The channel width CW represents a "low-stress" routing situation that usually occurs in commercial FPGAs for 'average' circuits.

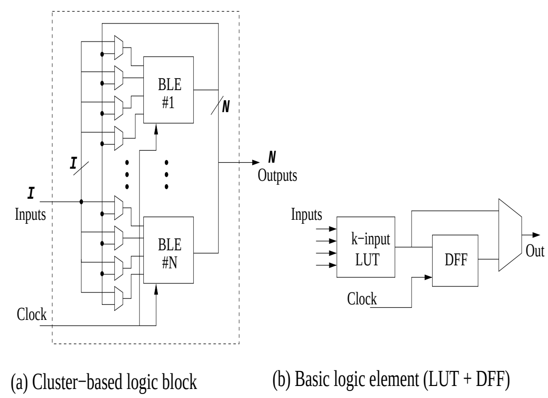
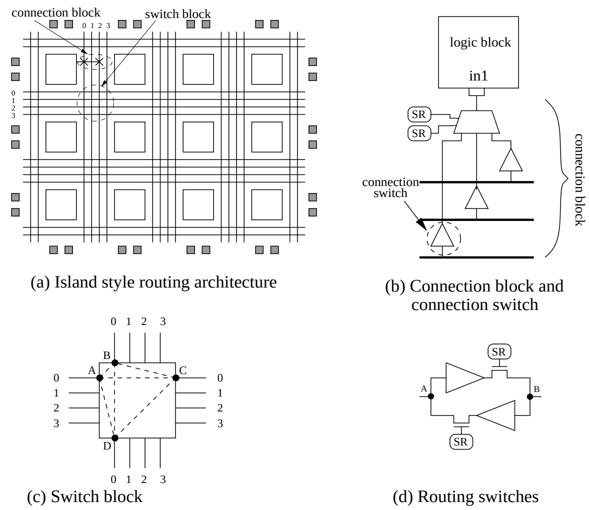


Fig. 1. FPGA logic block and basic logic element.

B. Evaluation Framework

This paper uses fpgaEVA-LP2 [3], [14] as the evaluation framework. fpgaEVA-LP2 includes a *BC-netlist* generator and a cycle-accurate power simulator. The BC-netlist generator takes the VPR placement and routing result and generates the Basic Circuit netlist (BC-netlist) annotated with post-layout capacitance and delay. The power simulator then performs cycle-accurate simulation on the BC-netlist to obtain FPGA power consumption. There are three power sources in FPGAs, switching power, short-circuit power and static power. The first two types of power contribute to the dynamic power and can only occur when a signal transition happens at the gate output. There are two types of signal transitions. *Functional transition* is the necessary signal transition to perform the required logic functions between two consecutive clock ticks. *Spurious transition* or *glitch* is the unnecessary signal transition due to the imbalanced path delays to the inputs of a gate. Glitch power can be a significant portion of the dynamic power. The short-circuit power is consumed when both PMOS and NMOS transistors are turned on in a gate. The third type of power, static power, is the power consumed when there is no signal transition for a gate or a circuit element.

[1] Without loss of generality, we assume subset switch block in this paper.
[2] We interchangeably use the terms of switch and tri-state buffer.



(a) Island style routing architecture

(b) Connection block and connection switch

(c) Switch block

(d) Routing switches

Fig. 2. (a) Island style routing architecture; (b) Connection block; (c) Switch block; (d) Routing switches.

In fpgaEVA-LP2, the power including switching power, short-circuit power and static power for logic blocks is pre-calculated per switch or per unit time by SPICE simulation, and so is the leakage power for interconnects. The interconnect switching power is calculated by a switch-level model with extracted parasitics, and its short-circuit power is calculated as a portion of switching power. This portion can be pre-calculated by SPICE simulation for a variety of input signal transition time and load capacitance.

In this paper, we use Berkeley predictive device model [15] and ITRS predictive interconnect model [16] at 100nm technology node. Table I summarizes the values of the key model parameters used throughout the rest of the paper. We use five small circuits from the MCNC benchmark set to illustrate the accuracy and fidelity of fpgaEVA-LP2 compared to SPICE simulation. The five circuits are chosen so that the circuit size is within the capability of SPICE simulation. They are mapped into 4-LUTs and packed into clusters with a cluster size of four. The largest circuit occupies six clusters and the smallest circuit occupies two clusters. As shown by the comparison in Figure 3, fpgaEVA-LP2 achieves high fidelity as well as high accuracy. The average of absolute error is 8.26% for the five test circuits [14].

Device model			
	Vdd (V)	NMOS-Vt (V)	PMOS-Vt (V)
normal-Vt	1.3	0.2607	-0.3030
high-Vt	1.3	0.4693	-0.5454
Interconnect model			
wire width	wire spacing	wire thickness	dielectric const.
0.56um	0.52um	1.08um	2.7

TABLE I

DEVICE AND INTERCONNECT MODEL AT 100NM TECHNOLOGY.

C. Evaluation Methodology and Results for Baseline Architecture Class

Our architecture evaluation starts with VPR placement and routing results. For a given FPGA architecture and benchmark circuit, VPR can generate different placement and routing results by using different seeds in its placement algorithm. Figure 4 shows the FPGA energy and delay using ten different VPR seeds for the same circuit s38584. We label the seed value beside each data point. The delay variation is 12% and the energy variation is 5%. This variation due to VPR seeds may affect our architecture evaluation. Because the delay variation is more sensitive to the VPR seeds than the energy variation, we decide to use the min-delay solution among all VPR seeds for every benchmark circuit. Note that the min-delay solution often consumes low energy too. For the architecture evaluation in this paper, Energy (E), Delay (D), Energy-Delay Product (ED) and Area (A) are always the geometric means of those values over 20 MCNC benchmark circuits in Table II.

Using the above methodology, we perform an architecture evaluation for the single-Vdd dual-Vt FPGA architectures from [8], defined as FPGA architecture *Class0* in this paper. The entire FPGA uses the uniform supply voltage 1.3V, but high-Vt is applied to all the FPGA configuration SRAM cells to reduce SRAM leakage power. The high-Vt configuration cells do not incur runtime performance degradation because they are constantly in read status after an FPGA is configured, and their read and write operations are irrelevant to the runtime performance. This high-Vt SRAM technique has already been used in commercial FPGAs and therefore we apply it to all FPGA architectures in this paper.

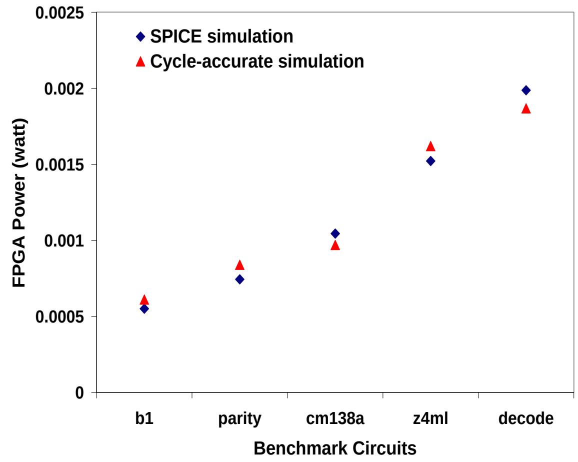


Fig. 3. Comparison between SPICE simulation and cycle-accurate power simulation.

circuit	# of nets	# of logic blocks	# of I/O blocks
alu4	782	162	22
apex2	1246	213	41
apex4	849	134	28
bigkey	1542	294	426
clma	7995	1358	144
des	1325	218	501
diffeq	1291	195	103
dsip	1139	588	426
elliptic	2617	666	245
ex1010	3033	513	20
ex5p	834	194	71
frisc	3240	731	136
misex3	828	181	28
pdc	2933	624	56
s298	908	66	10
s38417	5426	982	135
s38584	4502	1046	342
seq	1138	274	76
spla	2091	461	122
tseng	918	305	174

TABLE II
STATISTICS OF MCNC BENCHMARK CIRCUITS ($N = 10, k = 4$).

Figure 5 presents the evaluation results for single-Vdd dual-Vt FPGA Class0. Each data point in the figure is an FPGA architecture represented by a tuple (N, k), where $N \in \{6, 8, 10, 12\}$ is the cluster size and $k \in [3, 7]$ is the LUT size. If one architecture (N_1, k_1) has smaller delay and less energy consumption than another architecture (N_2, k_2), we say that architecture (N_1, k_1) is superior to (N_2, k_2). We define *strictly energy-delay dominant architectures* as the set of superior data points in the entire energy-delay tradeoff space. Those architectures are highlighted by the polyline in Figure 5. Our results also show that some of the architectures may have fairly similar energy and delay such as architectures $(N = 8, k = 4)$, $(N = 6, k = 4)$ and $(N = 10, k = 4)$, and all of of them can be valid solutions in reality. To avoid pruning out architectures with slightly worse energy and delay, we further define *relaxed energy-delay dominant architectures*. If architectures (N_1, k_1) and (N_2, k_2) have both energy and delay difference less than r (*relaxation parameter*), then neither of them can dominate the other one. With $r = 2\%$ in this paper, the relaxed dominant architectures are data points inside the highlighted area in Figure 5. Min-delay and min-energy architectures are the two extreme cases among those energy-delay dominant architectures. The min-delay architecture is $(N = 8, k = 7)$ and the min-energy architecture is $(N = 8, k = 4)$ for the FPGA Class0 in Figure 5, and the energy and delay differences between the two extreme cases are 57% and 14%, respectively. It shows that a significant tradeoff between energy and delay can be obtained by varying cluster size and LUT size. Note that our min-energy architecture $(N = 8, k = 4)$ is also the min-area architecture found by [10]. Commercial FPGAs such as Xilinx Virtex-II [17] coincidently use a cluster size of 8 and an LUT size of 4, and therefore their architectures may have used min-area solution and turn out to be a min-energy architecture in single-Vdd architecture class.

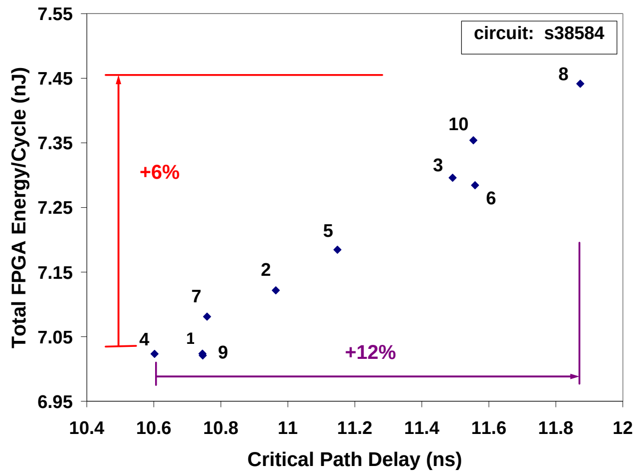


Fig. 4. Impact of random seed on FPGA energy and delay.

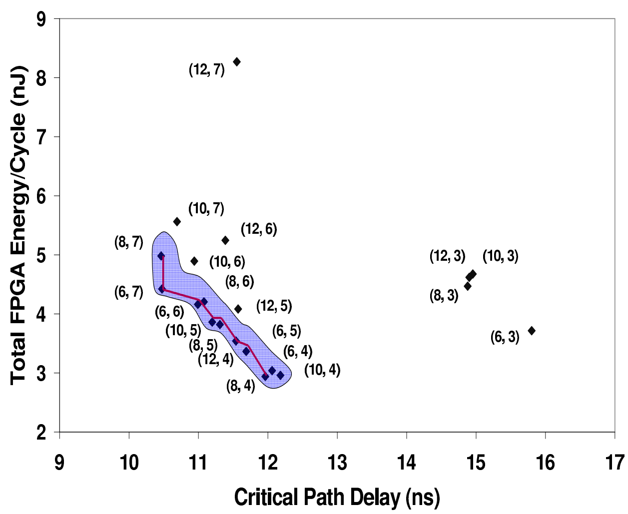


Fig. 5. Energy-delay tradeoff for single-Vdd dual-Vt FPGA class (Class0). The polyline represents the strictly dominant architectures and the enclosed area covers the relaxed dominant architectures.

D. Field Programmability of Vdd Supply vs. Pre-determined Vdd Pattern

It is well known that higher supply voltage leads to higher performance but larger power. Leveraging this, Vdd scaling lowers the supply voltage to the entire design or a large circuit module for power reduction. Alternatively, dual-Vdd applies high supply voltage (VddH) to logic on critical path and low supply voltage (VddL) to logic not on critical path. For given performance constraints, dual-Vdd is able to achieve more power reduction than Vdd scaling for ASICs [18]. Dual-Vdd or multi-Vdd technique has been successfully employed in ASICs [19]–[21] and an optimized multi-Vdd system can achieve dynamic power reductions of roughly 40-45% [22], [23]. The success of dual-Vdd is due to the fact that the designer is able to customize Vdd pattern for different applications.

Following the successful application of dual-Vdd techniques in ASICs, FPGAs might also benefit from those techniques for power reduction. However, there are some unique challenges to apply dual-Vdd to FPGAs. FPGAs do not have the freedom of using mask patterns to arrange different Vdd components in a flexible way as ASICs [18], [24]. Pre-defined dual-Vdd FPGA fabric may limit the power reduction by dual-Vdd techniques.

Assuming a generic cluster-based FPGA architecture [11] and MCNC benchmark circuit s38584, the power and performance curve for both Vdd scaling and dual-Vdd proposed in [8] is shown in Figure 6. The Vdd-level is decided uniformly for all logic blocks in Vdd scaling. Furthermore, each logic block has a pre-determined Vdd-level in dual-Vdd, and various Vdd patterns are tried to obtain the best result. It is easy to see from this figure that dual-Vdd consumes more power than Vdd scaling for a given frequency. Such power inefficiency is due to a pre-determined Vdd pattern, which imposes placement constraints and increases interconnect delay (and power). In contrast, the power and performance curve using programmable dual-Vdd logic blocks proposed in [9] is also presented in this figure. The programmable dual-Vdd reduces power significantly compared to

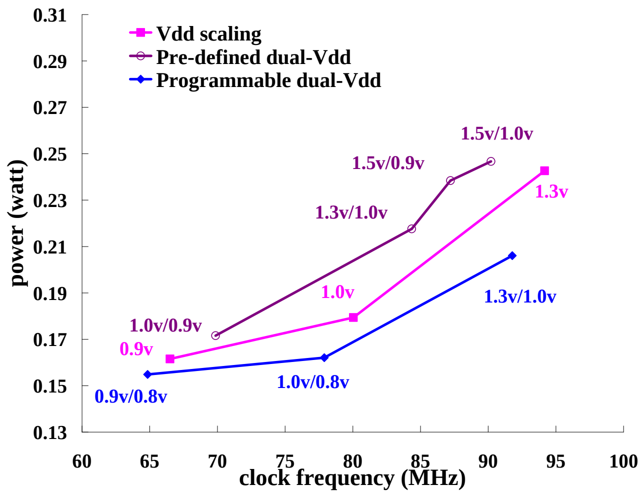


Fig. 6. Comparison of three power reduction solutions for benchmark s38584.

Vdd scaling. Field programmability of power supply is required to achieve FPGA power reduction via dual-Vdd. In this paper, we only study Vdd-programmable interconnects.

III. FPGA Circuits for Vdd Programmability

A. Previous Work and Section Overview

Programmable dual-Vdd has been introduced in [8], [9] and applied to logic blocks to reduce FPGA power. We define Vdd programmability as the flexibility to select Vdd levels for used circuit elements and the capability to power-gate unused circuit elements. Figure 7 shows the Vdd-programmable logic block. Two extra PMOS transistors, called *power switches* or *power transistors*, [3] are inserted between the conventional logic block and the dual-Vdd power rails for Vdd selection and power-gating.

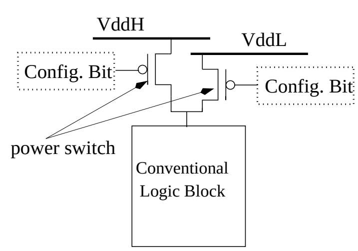


Fig. 7. Vdd-programmable logic block.

Power transistors usually use high-Vt for better leakage reduction in power-off state. Transistors with high-Vt have larger on-resistance and increase area for the specified performance. We use normal-Vt power transistors with gate-boosting same as those in [9] to reduce area overhead and achieve effective leakage reduction. When power-gating an unused logic block, the gate voltage of a PMOS power transistor is driven to one Vt higher than the Vdd at its source node. It has been shown that a gate-boosted power transistor can reduce leakage by two orders of magnitude compared to a normal transistor [9]. Note that gate-boosting has already been used in some commercial Xilinx FPGAs [11] to compensate the logic '1' degradation of NMOS pass transistors in routing switches. Therefore, it is not difficult to implement gate-boosting for our PMOS power transistors and achieve the same effective leakage reduction as high-Vt power transistors. In this paper, we use 210X minimum width PMOS power transistor for logic blocks ($N = 10, k = 4$) with delay overhead bounded by 5%.

In this section, we further apply Vdd programmability to interconnect switches. Normal-Vt power transistors with gate-boosting are used for interconnect switches. We design two types of interconnect switches, Vdd-programmable switch and Vdd-gateable switch. A Vdd-programmable switch provides three power states which are VddH, VddL and power-gating. Different from a Vdd-programmable switch, a Vdd-gateable switch only provides two power states between a pre-determined Vdd and power-gating, but it can dramatically reduce the number of configuration SRAM cells for Vdd programmability. The detailed circuit level designs of Vdd-programmable and Vdd-gateable switches are discussed below.

[3]The terms power switch and power transistor are used interchangeably in this paper.

B. Vdd-programmable Interconnect Switch

Figure 8 shows the design of Vdd-programmable interconnect switches (both routing switch and connection switch). A Vdd-level converter is needed whenever a VddL interconnect switch drives a VddH interconnect switch. In other cases, the level converter can be bypassed. As shown in Figure 8 (a), a pass transistor M1 and a MUX together with a configuration SRAM cell can be used to implement a configurable level conversion. The transistor M1 is used to prevent signal transitions from propagating through the level converter when it is bypassed, and therefore eliminate the dynamic power of an unused level converter. Only one configuration bit is needed to realize the level converter selection and signal gating for an unused level converter. The same asynchronous level converter circuit in [8], [9] is used to achieve a bounded delay with minimum power consumption.

For Vdd-programmable routing switch in Figure 8 (b), two PMOS power transistors M3 and M4 are inserted between the tri-state buffer and VddH, VddL power rails, respectively. Turning off one of the power transistors can select a Vdd level for the routing switch. By turning off both power transistors, an unused routing switch can be power-gated. The pass transistor M2 must be kept to prevent sneak path [25], i.e., a current path that flows from Vdd to ground through a set of "on" transistors which belong to different gates. SPICE simulation shows that power-gating the routing switch can reduce leakage power of a routing switch by a factor of over 1000. There are power and delay overhead associated with the power transistors insertion. As shown in Table III, the dynamic power overhead is almost negligible. This is because that the power transistors stay either ON or OFF after configuration and there is no charging and discharging at their source/drain capacitors. The delay overhead associated with the power transistor insertion can be bounded when the power transistor is properly sized. Another type of routing resources is the connection block in Figure 8 (c). The multiplexer-based implementation chooses only one track in the routing channel and connects it to the logic block input pin. The buffers between the routing tracks and the multiplexer are connection switches. Similar to the routing switch, programmable-Vdd is also applied to the connection switch. The multiplexer must be kept to prevent sneak path.

Vdd	routing switch delay (ns)		energy per switch (Joule)	
	w/o Vdd programmability	w/ Vdd programmability (increase %)	w/o Vdd programmability	w/ Vdd programmability
1.3v	5.90E-11	6.86E-11 (+16.27%)	3.3049E-14	3.2501E-14
1.0v	6.45E-11	7.55E-11 (+17.05%)	1.6320E-14	1.6589E-14

TABLE III

THE DELAY AND POWER OF A VDD-PROGRAMMABLE ROUTING SWITCH. WE USE 7X MINIMUM WIDTH TRI-STATE BUFFER FOR ROUTING SWITCHES AND 4X MINIMUM WIDTH PMOS TRANSISTOR FOR POWER TRANSISTORS.

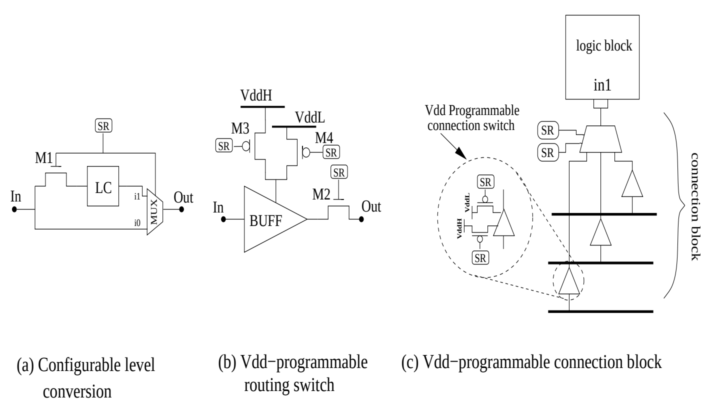


Fig. 8. (a) Configurable level conversion; (b) Vdd-programmable routing switch; (c) Vdd-programmable connection block. (SR stands for SRAM cell and LC stands for level converter.)

There are three SRAM cells for each Vdd-programmable routing switch in Figure 8 (b). For a connection block containing N Vdd-programmable connection switches in Figure 8 (c), there are $2N + \lceil log_2 N \rceil$ configuration SRAM cells, among which $\lceil log_2 N \rceil$ SRAM cells are for multiplexer and the other $2N$ extra SRAM cells are for N Vdd-programmable connection switches. We can use combinational logic such as decoder to reduce the number of extra SRAM cells introduced by Vdd programmability. As shown in Figure 9 (a), We first define a Vdd-programmable switch module with three signal ports, $VddH\_En$, $VddL\_En$ and $Pass\_En$. By setting these three control signals, we can program Vdd-programmable switch between Vdd selection and power-gating.

We design a new Vdd-programmable routing switch in Figure 9 (b). $Pass\_En$ can be generated by $VddH\_En$ and $VddL\_En$ with a NAND2 gate. Table IV summarizes the configurations for Vdd-programmable routing switch and the truth table of the relevant control signals.

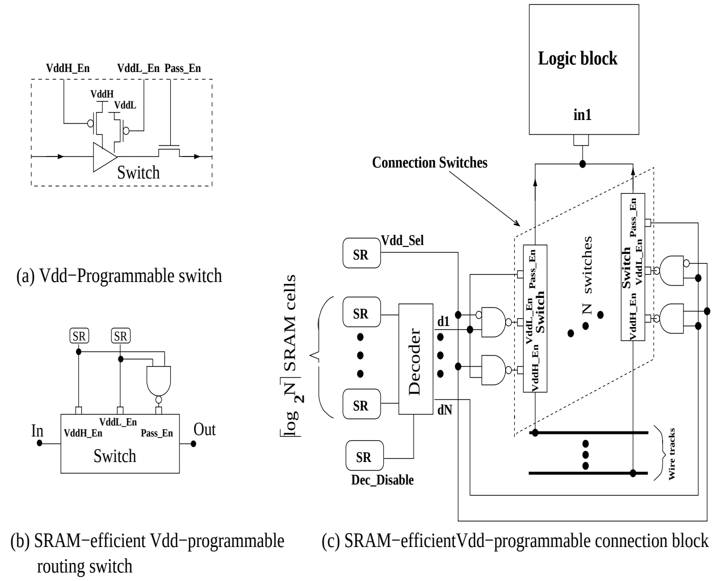


Fig. 9. (a) Vdd-programmable switch (b) SRAM-efficient Vdd-programmable routing switch; (c) SRAM-efficient Vdd-programmable connection block.

state	$VddH\_En$	$VddL\_En$	$Pass\_En$
VddH	0	1	1
VddL	1	0	1
power-gated	1	1	0

TABLE IV

CONFIGURATIONS FOR A VDD-PROGRAMMABLE ROUTING SWITCH.

Similarly, Figure 9 (c) presents a new design of Vdd-programmable connection block with reduced configuration SRAM cells. For a connection block containing N connection switches, we use a $\lceil log_2 N \rceil : N$ decoder and $2N$ NAND2 gates as the control logic. There is a disable signal $Dec\_Disable$ for the decoder. Each decoder output is connected to $Pass\_En$ of one connection switch. Setting $Pass\_En$ of a connection switch to '0' can power-gate this switch by setting both $VddH\_En$ and $VddL\_En$ to '1' with NAND2 gates. When the whole connection block is not used, all N outputs of the decoder are set to '0' to power-gate all the connection switches by asserting $Dec\_Disable$. When the connection block is in use, $Dec\_Disable$ is not asserted. By using $\lceil log_2 N \rceil$ configuration bits for the decoder, only one $Pass\_En$ is set to '1' and others are set to '0', i.e., only one connection switch inside the connection block is selected and connects the one track to the logic block input, and other unused connection switches are power-gated. Another configuration bit Vdd_Sel is used to select the Vdd level for the selected connection switch. Table V summarizes configurations for Vdd-programmable connection switch and the truth table of relevant control signals.

state	$Dec\_Disable$	Vdd_Sel	$Pass\_En$	$VddH\_En$	$VddL\_En$
power-gated	1	-	0	-	-
power-gated	0	-	0	-	-
VddH	0	1	1	0	1
VddL	0	0	1	1	0

TABLE V

CONFIGURATIONS FOR A VDD-PROGRAMMABLE CONNECTION SWITCH.

By replacing the conventional connection switch with the new Vdd-programmable switch in Figure 9 (a), the pass transistor in the Vdd-programmable switch can now prevent sneak path. Therefore, the multiplexer implemented by NMOS pass transistor tree can be removed from the new Vdd-programmable connection block. Table VI shows the delay and power of a new Vdd-programmable connection block. The delay and dynamic energy per signal transition are reduced by 28% and 19% respectively when Vdd-level is 1.3v. The delay and power reduction is due to multiplexer removal.

For a connection block containing N connection switches, only $\lceil log_2 N \rceil + 2$ configuration SRAM cells are needed to provide Vdd selection and power-gating capability for each individual connection switch inside the connection block. Compared to a conventional connection block, only two extra configuration SRAM cells are introduced. Similar to the SRAM cell, we use high-Vt transistors for control logic to reduce leakage overhead as the delay of control logic will not affect system runtime performance. We also use minimum width transistors for control logic to reduce area overhead. In this paper,we use the same area model as that in [11], in which the area is counted in number of minimum width transistor areas with considering the

Vdd	connection switch delay (ns)		energy per switch (Joule)	
	w/o Vdd program-mability	w/ Vdd programmability (increase %)	w/o Vdd program-mability	w/ Vdd programmability (increase %)
1.3v	2.93E-10	2.10E-10 (-28.33%)	3.84E-14	3.11E-14 (-19.01%)
1.0v	3.70E-10	2.22E-10 (-40.00%)	3.09E-14	2.04E-14 (-33.98%)

TABLE VI

THE DELAY AND POWER OF NEW VDD-PROGRAMMABLE CONNECTION BLOCK. WE USE 4X MINIMUM WIDTH TRI-STATE BUFFER FOR CONNECTION SWITCHES AND 1X MINIMUM WIDTH PMOS TRANSISTOR FOR POWER TRANSISTORS.

parallel diffusions technique for large transistors. Given a transistor with channel width W, the transistor area measured by the minimum width transistor with channel width W_{min} is:

$$Area(W) = 0.5 + \frac{W}{2 \cdot W_{min}}. \tag{1}$$

Table VII compares the number of configuration SRAM cells, leakage and area between Vdd-programmable routing switches/connection block in Figure 8 and Figure 9, respectively. The Vdd-programmable routing switch and connection block in Figure 9, called *SRAM-efficient switches*, have smaller area and less leakage, and will be used in the rest part of the paper.

Vdd-programmable routing switch			SRAM-efficient Vdd-programmable routing switch					compared to baseline: w/o control logic		
SRAM cells			SRAM cells			NAND2				
number	leakage (watt)	area	number	leakage (watt)	area	leakage (watt)	area	Δ number of SRAM cells	Δ leakage (watt)	Δ area
3	2.32E-8	21.87	2	1.55E-8	14.58	3.49E-10	2.50	-1	-7.38E-9	-4.79
32:1 connection block										
Vdd-programmable connection block			SRAM-efficient Vdd-programmable connection block					compared to baseline: w/o control logic		
SRAM cells			SRAM cells			control logic				
number	leakage (watt)	area	number	leakage (watt)	area	leakage (watt)	area	Δ number of SRAM cells	Δ leakage (watt)	Δ area
69	5.32E-7	503.01	7	5.42E-8	43.74	3.30E-8	311	-62	-4.56-E7	-148.27

TABLE VII

THE COMPARISON OF THE NUMBER OF SRAM CELLS, LEAKAGE AND AREA BETWEEN A VDD-PROGRAMMABLE ROUTING SWITCH/CONNECTION BLOCK AND AN SRAM-EFFICIENT VDD-PROGRAMMABLE ROUTING SWITCH/CONNECTION BLOCK. WE USE 32:1 CONNECTION BLOCK AND THE CONTROL LOGIC FOR SRAM-EFFICIENT DESIGN CONTAINS A STANDARD 5:32 DECODER AND 64 NAND2 GATES. AREA IS PRESENTED IN MINIMUM WIDTH TRANSISTOR AREAS.

C. Vdd-gateable Interconnect Switch

Compared to Vdd-programmable switch, Vdd-gateable interconnect switch only provides two power states between a pre-determined Vdd level and power-gating, but it can dramatically reduce the number of extra SRAM cells for Vdd programmability. Figure 10 (a) shows the circuit design for a Vdd-gateable switch. Based on a conventional tri-state buffer, we insert a PMOS transistor M2 between the power rail and the tri-state buffer to provide the power-gating capability. When a switch is not used, transistor M1 is turned off by the configuration cell SR. At the same time, we can turn off M2 to power-gate the unused switch. Similarly, both M1 and M2 are turned on by the configuration cell SR when the switch is used. Thus, we do not need to introduce an extra SRAM cell for power-gating capability. Figure 10 (b) presents Vdd-gateable routing switches. We can reduce leakage power by a factor of over 1000 for an unused switch when it is power-gated. Similar to the Vdd-programmable switch, the pass transistor M1 must be kept to prevent sneak path. However, there is a delay overhead associated with the M2 insertion. We properly size M2 for the tri-state buffer to achieve a delay increase bounded by 6%. Similar to Vdd-programmable switch, dynamic power overhead associated with the insertion of PMOS M2 is almost negligible because transistor M2 is always ON when the routing switch is used and there is no charging or discharging occur at its source/drain capacitors.

The design of Vdd-gateable connection block is shown in Figure 10 (c). We only need $\lceil log_2 N \rceil$ configuration SRAM cells to control N connection switches in a connection block via a decoder with complementary outputs and achieve the power-gating capability for each connection switch at the same time. We use another configuration bit, $Dec\_Disable$, to disable the decoder when we apply power-gating to the whole connection block. Similar to the SRAM-efficient design of Vdd-programmable switch, we use high-Vt and minimum width transistor for the decoder to reduce leakage and area overhead. Alternatively, N configuration SRAM cells can be used to control the same number of connection switches without using the decoder. Table VIII compares the number of SRAM cells, leakage and area for a non-decoder based and decoder based connection block containing 32 connection switches. The decoder based Vdd-gateable connection block consumes less area and leakage power, and will be used in the rest part of this paper.

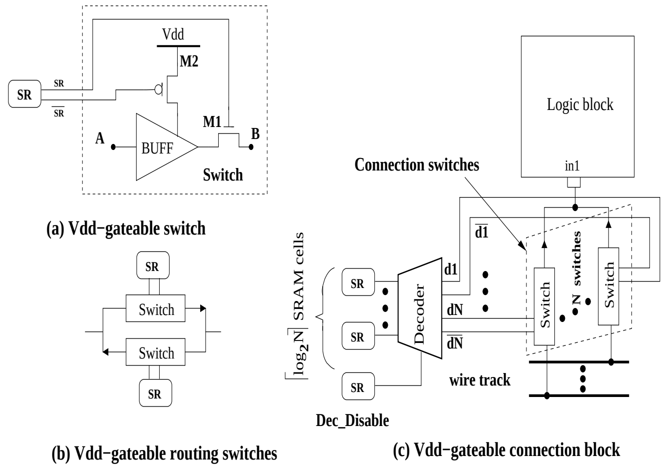


Fig. 10. (a) Vdd-gateable switch; (b) Vdd-gateable routing switches; (c) Vdd-gateable connection switches. (SR stands for SRAM cell)

Comparison between non-decoder based and decoder based 32:1 Vdd-gateable connection block										
non-decoder based connection block			decoder based connection block					compared to baseline: w/o decoder		
SRAM cells			SRAM cells			5:32 decoder				
number	leakage (watt)	area	number	leakage (watt)	area	leakage (watt)	area	Δ number of SRAM cells	Δ leakage (watt)	Δ area
32	2.47E-7	233.28	6	4.63E-8	43.74	2.00E-8	94.25	-26	-1.81E-7	-95.29

TABLE VIII

THE COMPARISON OF THE NUMBER OF SRAM CELLS, LEAKAGE AND AREA BETWEEN A NON-DECODER BASED VDD-GATEABLE CONNECTION BLOCK AND A DECODER BASED VDD-GATEABLE CONNECTION BLOCK. WE USE A 32:1 CONNECTION BLOCK. FOR THE DECODER BASED VDD-GATEABLE CONNECTION BLOCK, WE USE A 5:32 DECODER WITH COMPLEMENTARY OUTPUT. AREA IS PRESENTED IN MINIMUM WIDTH TRANSISTOR AREAS.

IV. ARCHITECTURE EVALUATION FOR VDD-PROGRAMMABLE FPGAS

In this section, we first evaluate two architecture classes *Class1* and *Class2* for Vdd-programmable FPGAs. Class1 applies programmable dual-Vdd to each logic block and each interconnect segment, and inserts a configurable level conversion circuit in front of each routing/connection switch as well as at the inputs/outputs of the logic blocks. Class2 applies programmable dual-Vdd only to logic blocks, and uses Vdd-gateable routing/connection switches in FPGA interconnects. Therefore, the interconnect switches in architecture Class2 only have two configurable states: high Vdd (VddH) and power-gating. As we use VddH for interconnects in architecture Class2, level converters are only needed at the logic block outputs, but not at the logic block inputs nor in routing channels. Similar to the baseline architecture Class0 in II-C, the configuration SRAM cells in both architecture classes use the high-Vt SRAM design. All these architecture classes (with Class3 to be presented in Section V) are summarized in Table IX.

Architecture Class	Logic block	Interconnect
Class0 (baseline)	single Vdd	single Vdd
Class1	programmable dual-Vdd	programmable dual-Vdd w/ level converters
Class2	programmable dual-Vdd	VddH and Vdd-gateable
Class3	programmable dual-Vdd	programmable dual-Vdd w/o level converters

TABLE IX

SUMMARY OF BASELINE ARCHITECTURE CLASS AND VDD-PROGRAMMABLE ARCHITECTURE CLASSES (LC DENOTES THE LEVEL CONVERTER).

We apply a simple yet practical design flow similar to that in [9]. As shown in Figure 11, starting with a single-Vdd gate level netlist, we apply technology mapping and timing-driven packing [11] to obtain the single-Vdd cluster-level netlist. We then perform single-Vdd timing-driven placement and routing by VPR [11] and generate the basic circuit netlist (BC-netlist). We calculate power sensitivity $\Delta P/\Delta V_{dd}$, which is the power reduction by changing VddH to VddL, for each circuit element. The total power P includes both switching power P_{sw} and leakage power P_{lkg}. For each node i, we have switching power $P_{sw}(i) = 0.5 f_{clk} \cdot E_i \cdot C_i \cdot V_{dd}^2$, where E_i and C_i are transition density and load capacitance, and leakage power $P_{lkg} = I_{lkg}(V_{dd}) \cdot V_{dd}$. We pre-characterize I_{lkg} and device delay at each Vdd level using SPICE simulation. We assume that the transition density for each circuit element will not change when some circuit elements are assigned to VddL, and therefore we only need to calculate the power sensitivity for each circuit element once. A greedy algorithm is carried out for Vdd assignment considering iteratively updated timing slack (See Figure 12). Assuming that all the circuit elements are

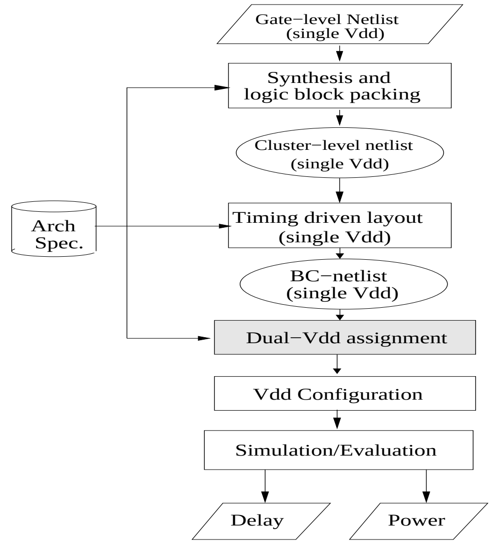


Fig. 11. Design flow for the fully Vdd-programmable FPGA fabric.

initially assigned to VddH, we iteratively perform the following steps. Timing analysis is performed to obtain the circuit elements on the path with the largest timing slack. We then assign VddL to the element with the largest power sensitivity. The configurable level converter can be enabled as needed. After updating the circuit timing, we accept the assignment if the critical path delay does not increase. Otherwise, we reject the assignment and restore the the circuit element supply voltage to VddH. In either case, the circuit element will be marked as 'tried' and will not be re-visited in subsequent iterations. After the dual-Vdd assignment, we obtain a dual-Vdd BC-netlist without degrading the system performance. For FPGA Class1, the Vdd assignment unit is a logic block or an interconnect switch. For FPGA Class2, the Vdd assignment unit is a logic block. For both Class1 and Class2, power-gating is applied to all unused logic blocks and programmable switches. Finally, we perform the energy and delay evaluation for the dual-Vdd design.

Sensitivity-based dual-Vdd assignment algorithm:
Assign VddH to all circuit elements and mark them as untried;
Calculate power-sensitivity S for all circuit elements;
While($\exists$ untried circuit elements)
{
  Assign VddL to the element with largest S if no critical path increase;
  Update timing slack and mark the element as tried;
}

Fig. 12. Sensitivity-based dual-Vdd assignment algorithm.

Figure 13 presents the energy-delay tradeoff in terms of different architectures, i.e., different combinations of cluster size N and LUT size k, for three FPGA classes: Class0, Class1 and Class2. Considering the VddL/VddH ratio between 0.6 $\sim$ 0.7 suggested in [23], we use 1.3v for VddH and 0.8v for VddL in our experiments. We only show the relaxed dominant architectures in the figure and the polylines represent the strictly dominant architectures. Similar to the baseline FPGA Class0, the min-delay architecture is $(N = 8, k = 7)$ for both Class1 and Class2. The min-energy architecture is $(N = 8, k = 4)$ for Class1 and $(N = 12, k = 4)$ for Class2. This shows that LUT size 7 gives the best performance and LUT size 4 leads to the lowest energy consumption for these Vdd-programmable FPGAs.

We then use the metrics of energy E, delay D and energy-delay product ED to compare the two classes of Vdd-programmable FPGAs (Class1 and Class2) and the baseline FPGA (Class0). We use the min-energy (min-delay) architecture within each FPGA architecture class and obtain the energy saving (delay increase) by Vdd-programmable FPGAs. As shown in Table X, FPGA Class1 obtains an energy saving of 28.57% and FPGA Class2 obtains an energy saving of 54.08% compared to the baseline architecture class. The delay increase due to Vdd programmability is only 3% for both FPGA Class1 and Class2. We also use the min-ED (i.e., the minimum energy-delay product) architecture within each architecture class and obtain the ED product reduction. FPGA Class1 reduces ED product by 25.97% and Class2 reduces ED product by 54.39%.

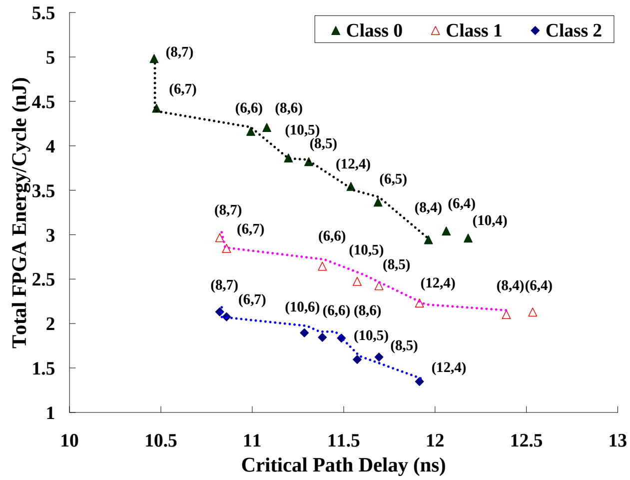


Fig. 13. Energy and delay tradeoff for the baseline single-Vdd dual-Vt FPGA (Class0) and the two classes of Vdd-programmable FPGAs (Class1 and Class2). The figure only shows relaxed energy-delay dominant solutions and the strictly dominant solutions are represented by polylines.

Arch. Class →	Class0 (baseline)	Class1	Class2	Class3
min-E arch. (N,k)	(8,4)	(8,4)	(12,4)	(12,4)
energy (nJ/cycle)	2.94	2.10	1.35	1.18
energy saving (%)	-	28.57%	54.08%	59.86%
min-D arch. (N,k)	(8,7)	(8,7)	(8,7)	(8,7)
delay (ns)	10.46	10.82	10.82	10.82
delay increase (%)	-	3%	3%	3%
min-ED arch. (N,k)	(8,4)	(8,4)	(12,4)	(12,4)
ED product (nJ · ns)	35.19	26.05	16.05	14.03
ED reduction	-	25.97%	54.39%	60.13%

TABLE X

COMPARISON BETWEEN VDD-PROGRAMMABLE FPGAS (CLASS1, CLASS2 AND CLASS3) AND THE BASELINE FPGA (CLASS0) USING ENERGY E, DELAY D AND ENERGY-DELAY PRODUCT (ED).

V. IMPROVED FPGA ARCHITECTURES

A. FPGA Architectures and Related CAD Algorithm

By using Vdd-programmable interconnects, we can reduce the interconnect dynamic energy which is not available by Vdd-gateable interconnects. However, as presented in Section IV, fully Vdd-programmable FPGA architecture Class1 consumes more energy than FPGA architecture Class2 which uses Vdd-gateable interconnects. This is because of the leakage overhead of the large number of Vdd-level converters in routing channels, which provides Vdd programmability for each individual interconnect switch. To achieve better energy-delay tradeoff, we design an improved fully Vdd-programmable FPGA architecture *Class3*. It uses the same SRAM-efficient interconnect switches as FPGA architecture Class1, but inserts level converters only at logic block inputs and outputs. Since there is no level converter in routing channels, we need a CAD algorithm to guarantee that no VddL interconnect switch drives VddH interconnect switch. We tackle the problem by choosing routing tree as the Vdd assignment unit. Similar to FPGA Class1, the same design flow and the sensitivity-based Vdd level assignment algorithm in Figure 12 is used to decide the Vdd level for each interconnect routing tree. The only difference is that we use an interconnect routing tree as the assignment unit for FPGA Class3 while an interconnect switch is used as the assignment unit for Class1. Since two routing trees will not intersect with each other in routing channels, we do not need level converters in routing channels. Figure 14 illustrates the situation that a VddH routing tree and VddL routing tree can share a same routing track without level converters in routing channels.

B. Energy and Delay Evaluation

In this section, we evaluate the improved fully Vdd-programmable architecture *Class3*. Figure 15 shows the energy-delay evaluation for the improved architecture Class3 compared to the evaluation results for architecture Class0, Class1 and Class2. As shown in Figure 15, we can see that the improved architecture Class3 can achieve better energy-delay tradeoff than architecture Class1, and even better than Class2. This is because FPGA Class3 removes the level converters in routing channels, but still

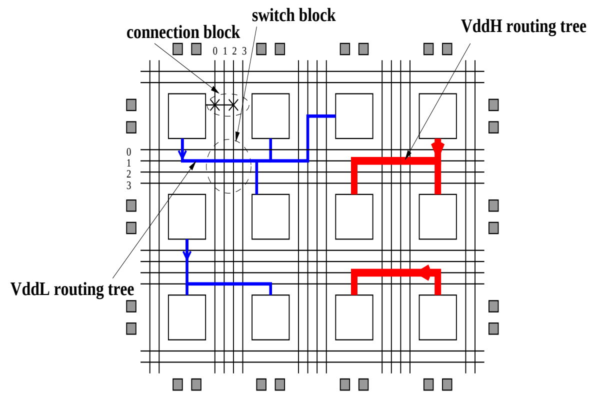


Fig. 14. Improved fully Vdd-programmable FPGA architecture Class3. No level converter is inserted in routing tracks.

can reduce interconnect dynamic energy. This is not available in architecture Class2 which uses Vdd-gateable interconnect switches.

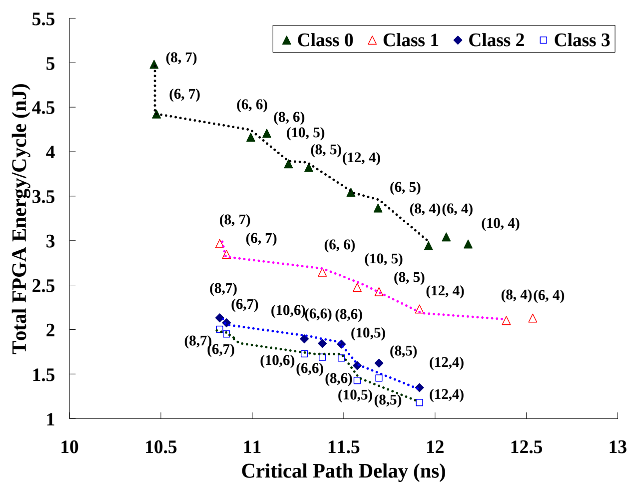


Fig. 15. Energy and delay tradeoff for all FPGA architecture classes. The figure only shows relaxed energy-delay dominant solutions and the strictly dominant solutions are represented by polylines.

Similar to Class0, the min-delay architecture is $(N = 8, k = 7)$ for Class3. The min-energy architecture is $(N = 12, k = 4)$ for Class3. $(N = 12, k = 4)$ also gives the minimum energy delay product ED in architecture Class3. We can see that for our improved FPGA architecture Class3, again, LUT size 7 gives the best performance and LUT size 4 leads to the lowest energy consumption. Compared to the min-energy (min-delay) architecture within baseline architecture Class0, the min-energy architecture in Class3 obtains an energy reduction of 59.86%, and the min-delay architecture in Class3 has a 3% delay overhead due to Vdd programmability. The min-ED architecture in FPGA Class3 reduces energy delay product ED by 60.13%. As shown in Table X, FPGA Class3 gives the lowest energy as well as the lowest energy delay product ED.

C. Energy and Area Evaluation

Figure 16 presents the energy-area curve for baseline FPGA Class0 and Vdd-programmable FPGA Class1, Class2 and Class3. The total device area includes both logic block and interconnect device area. The area overhead of extra configuration SRAM cells, power transistors and Vdd-level converters are included. In this figure, we show the relaxed ED-dominant architectures as well as the min-area architecture in each FPGA class. The polylines represent the lowest energy-area envelop in each class. The min-area architecture is $(N = 8, k = 4)$ for Class0 and Class1, $(N = 10, k = 4)$ for Class2 and Class3. We can see that LUT size 4 not only gives the lowest energy consumption, but also gives the minimum area. In FPGA Class2 and Class3, the min-area architecture $(N = 10, k = 4)$ consumes similar energy with the min-energy architecture $(N = 12, k = 4)$ while it gives much smaller total device area.

Vdd programmability increases the total number of SRAM cells required to store those extra configuration bits. However, SRAM cells are vulnerable to soft errors and the total number of SRAM cells should be minimized. Table XI presents the

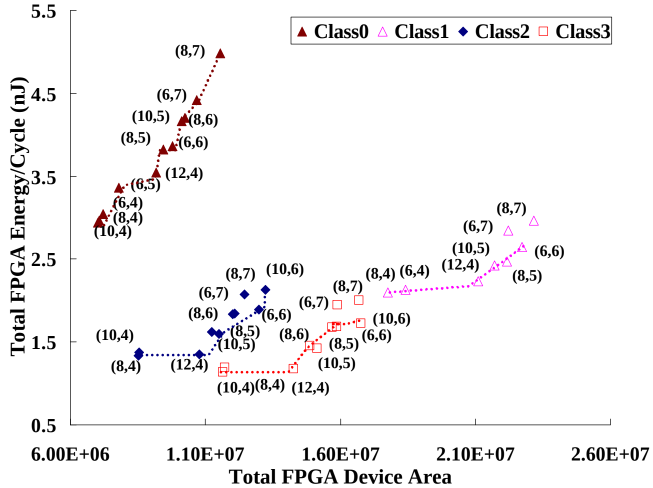


Fig. 16. Energy versus area curve for all architecture classes. This figure only shows relaxed energy-delay dominant solutions and min-area solution within each FPGA class. The polylines represent the lowest energy-area envelop. Area is measured in minimum width transistor areas.

increase in SRAM cell number and the total device area overhead due to Vdd programmability. The SRAM cells include those used in LUTs. Only dominant architectures are shown in the table. Vdd-programmable FPGA Class1 increases the SRAM cell number by 132%. This shows that fully Vdd-programmable FPGAs need a large number of extra SRAM cells to provide fine-grained Vdd programmability for interconnects. FPGA Class2 only increases the SRAM cell number by 3% because only two power states (VddH and power-gating) are provided for FPGA interconnect switches and the original SRAM cells for interconnection programmability can be shared for Vdd programmability. Compared to FPGA Class1, the improved FPGA Class3 using the same Vdd-programmable switches only increases the SRAM cell number by 28%. This is because FPGA Class3 applies Vdd programmability to each routing tree without any configurable level converters in routing channels. FPGA Class1 has 118% area overhead, FPGA Class2 has 17% area overhead and FPGA Class3 has 52% area overhead. Both Class2 and Class3 introduce less SRAM and area overhead while reducing more energy compared to Class1. FPGA Class2 reduce comparable energy while it gives much smaller SRAM and area than Class3.

Dominant Arch. (N,k)	total # of SRAM cells on chip				total device area			
	Class0 baseline	Class1 (% overhead)	Class2 (% overhead)	Class3 (% overhead)	Class0 baseline	Class1 (% overhead)	Class2 (% overhead)	Class3 (% overhead)
(8,7)	649218	88%	2%	17%	11541440	100%	15%	44%
(6,7)	621929	89%	2%	20%	10689783	108%	16%	49%
(6,6)	469504	128%	3%	31%	10114162	125%	19%	57%
(10,5)	374174	164%	3.4%	33%	9793576	126%	17%	55%
(12,4)	317391	190%	4%	40%	9173613	130%	17%	55%
Average	-	132%	3%	28%	-	118%	17%	52%

TABLE XI

TOTAL NUMBER OF CONFIGURATION SRAM CELLS AND DEVICE AREA OVERHEAD FOR DIFFERENT VDD-PROGRAMMABLE FPGAS. SRAM CELLS INCLUDE THOSE USED IN LUTS AND TOTAL DEVICE AREA INCLUDES BOTH LOGIC BLOCK AND INTERCONNECT AREA. THE DEVICE AREA IS IN MINIMUM WIDTH TRANSISTOR AREA.

D. Energy and Area Overhead Breakdown

Figure 17 presents the energy breakdown of architecture $(N = 12, k = 4)$ for all FPGA architecture classes. The logic energy is the energy of LUTs, flip-flops and MUXes in logic blocks. The local interconnect energy is the energy of internal routing wires and buffers within logic blocks. Routing wires outside logic blocks, programmable interconnect switches in routing channels and their configuration SRAM cells contribute to global interconnect energy. It is clear that both FPGA Class2 and FPGA Class3 can dramatically reduce global interconnect leakage energy due to the extremely low utilization rate of interconnect switches ($\sim$3% on average for architecture $(N = 12, k = 4)$ as shown in Table XII). FPGA Class1 fails due to large leakage overhead of Vdd-level converters in routing channels. The figure also shows that global interconnect dynamic energy, 59.24% of total FPGA energy for Class2 and 52.34% for Class3, becomes dominant after applying programmable Vdd technique.

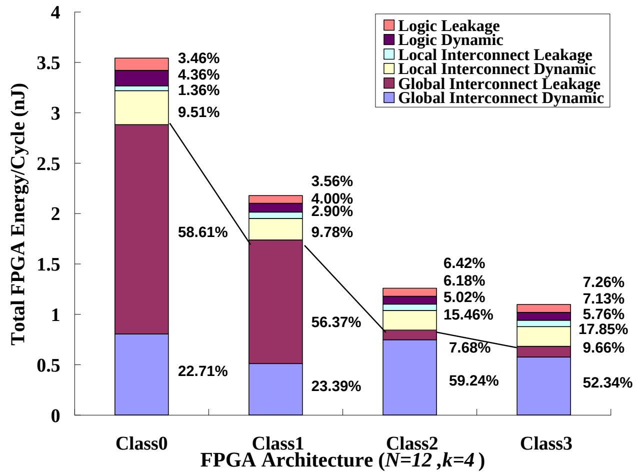


Fig. 17. Energy breakdown of architecture (12,4) for all classes.

circuit	routing switch			connection switch			interconnect switch		
	total #	used #	utilization	total #	used #	utilization	total #	used #	utilization
alu4	22843	2446	10.71%	99216	2421	2.44%	122059	4867	3.99%
apex2	48721	4835	9.92%	211484	3810	1.80%	260205	8645	3.32%
apex4	28598	3256	11.39%	124267	2293	1.85%	152865	5549	3.63%
bigkey	52464	3735	7.12%	227448	2475	1.09%	279912	6210	2.22%
clma	416040	36919	8.87%	1803360	20438	1.13%	2219400	57357	2.58%
des	92130	5291	5.74%	399360	3004	0.75%	491490	8295	1.69%
diffeq	25300	2953	11.67%	109850	2506	2.28%	135150	5459	4.04%
dsip	76510	3729	4.87%	331695	1995	0.60%	408205	5724	1.40%
elliptic	90888	8892	9.78%	394212	5635	1.43%	485100	14527	2.99%
ex1010	126912	12779	10.07%	550368	8587	1.56%	677280	21366	3.15%
ex5p	30046	3099	10.31%	130559	2089	1.60%	160605	5188	3.23%
frisc	158600	13886	8.76%	687700	8509	1.24%	846300	22395	2.65%
misex3	26722	3100	11.60%	116064	2503	2.16%	142786	5603	3.92%
pdc	181375	17304	9.54%	786500	9989	1.27%	967875	27293	2.82%
s298	40440	3553	8.79%	175500	3804	2.17%	215940	7357	3.41%
s38417	148648	14140	9.51%	644436	10282	1.60%	793084	24422	3.08%
s38584	127432	11910	9.35%	552500	8113	1.47%	679932	20023	2.94%
seq	36938	4278	11.58%	160381	3244	2.02%	197319	7522	3.81%
spla	109282	10329	9.45%	473993	6789	1.43%	583275	17118	2.93%
tseng	17738	2361	13.31%	77077	1847	2.40%	94815	4208	4.44%
Avg.	-	-	9.62%	-	-	1.61%	-	-	3.11%

TABLE XII

INTERCONNECT SWITCH UTILIZATION RATE OF FPGA ARCHITECTURE $(N = 12, k = 4)$.

Figure 18 presents the area overhead breakdown of architecture $(N = 12, k = 4)$ for FPGA architecture Class2 and Class3. The area overhead of routing switches and connection blocks is introduced by power transistors, extra configuration SRAM cells and control logics. The area overhead of logic blocks is introduced by Vdd-level converters at logic block inputs/outputs and associated configuration SRAM cells. The area overhead of FPGA Class2 due to routing switches, connection blocks and logic blocks are 3.87%, 11.31% and 1.95%, respectively. The area overhead of FPGA Class3 due to routing switches, connection blocks and logic blocks are 16.93%, 34.22% and 3.19%, respectively. The area overhead due to connection blocks is dominant for both FPGA Class2 and Class3.

From another point of view, the area overhead of FPGA Class2 due to power transistors and control logics are 10.22% and 4.82%, repspectively. The area overhead due to extra configuration SRAM cells is less than 1% for FPGA Class2. For FPGA Class3, the area overhead due to power transistors, control logics and extra configuration SRAM cells are 19.05%, 25.47% and 8.02%, respectively. Power transistors introduce the largest area overhead for FPGA Class2 while control logics introduce the largest area overhead for FPGA Class3.

VI. CONCLUSIONS AND DISCUSSIONS

The existing FPGA designs are highly power inefficient compared to their ASIC counterparts. As the process advances to nanometer technology and low-energy embedded applications are explored for FPGAs, power consumption becomes a crucial design constraint for FPGAs.

As far as energy-efficient FPGAs are concerned, we have designed novel Vdd-programmable and Vdd-gateable interconnect switches with minimal number of configuration SRAM cells to reduce FPGA energy. Using the new switches, we have proposed three new classes of Vdd-programmable FPGA architectures. *Class1* applies Vdd programmability to each logic block and each interconnect segment, with a large number of Vdd-level converters inserted for fine-grained Vdd programmability in interconnects. *Class2* uses Vdd-programmable logic blocks and Vdd-gateable interconnects. Similar to *Class1*, *Class3* also

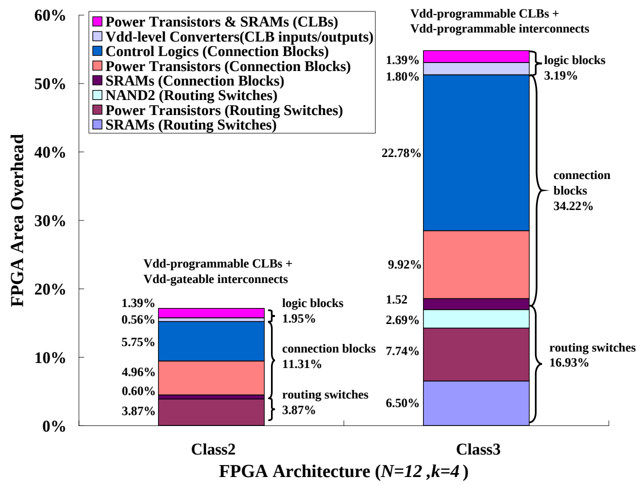


Fig. 18. Area overhead breakdown of architecture (12,4) for FPGA architecture Class2 and Class3.

applies Vdd programmability to both logic blocks and interconnects, but it applies Vdd programmability to each routing tree without any Vdd-level converter in routing channels. We have conducted FPGA architecture evaluation. The baseline for comparison is *Class0*, which uses high-Vdd for both logic blocks and interconnects. High-Vt is applied to configuration SRAM cells for all four architecture classes, and the same dual-Vdd levels are applied to Class1, Class2 and Class3. Using the metric of Energy-Delay Product (ED) measured as a geometric mean over the MCNC benchmark set, the ED reduction for the min-ED architecture in Class1, Class2 and Class3 is 25.97%, 54.39% and 60.13% respectively, when compared to Class0 which is equivalent to the cutting-edge commercial products. The SRAM cell overhead introduced by Vdd-programmability for Class1, Class2 and Class3 is 132%, 3% and 28%, respectively. The total device area overhead for Class1, Class2 and Class3 is 118%, 17% and 52%, respectively. Both FPGA Class2 and Class3 achieve more energy reduction with less SRAM and area overhead compared to FPGA Class1. While FPGA Class3 gives the lowest energy consumption, FPGA Class2 achieves comparable energy reduction with significantly reduced number of SRAM cells and device area overhead. We conclude that Class2 is the best architecture class considering area, power and performance tradeoff. Our evaluation results also show that, within each architecture class, LUT size 4 gives the lowest energy consumption as well as the smallest total device area while LUT size 7 leads to the highest performance.

Increased area due to Vdd programmability makes wire segment longer and wire capacitance per segment larger that will result in larger energy consumption. We do not consider longer wire segment due to larger chip size in our analysis. As 17% area overhead only leads to 8% longer wire segment and the load capacitance of a routing switch is usually dominated by its fanout routing switches in FPGA, we speculate that slightly less energy reduction can be achieved for FPGA Class2 considering this factor, and Class2 is still the best architecture class considering area, power and performance tradeoff.

There are a few alternative architecture classes. One alternative class may apply single-Vdd with power-gating to both logic blocks and interconnects. One configuration SRAM cell and one power transistor can be used to provide field power-gating capability for any circuit element, such as a logic block or an interconnect segment. However, the area overhead due to logic block Vdd programmability is small (∼3% for architecture $(N = 12, k = 4)$), and the utilization rate of logic blocks is high (∼83% for architecture $(N = 12, k = 4)$) as the smallest square FPGA array is used for each benchmark circuit. We speculate this FPGA class may reduce less but similar energy with smaller but similar area overhead compared to FPGA Class2, and have not evaluated this architecture class. Another alternative FPGA class may use one configuration SRAM cell and two power transistors for field programming the Vdd level (VddH or VddL) for any circuit element. In this case, VddL can be applied to the unused circuit elements to reduce leakage. Given the similar energy reduction between Class2 with Vdd-gateable interconnects and Class3 with Vdd-programmable (including Vdd-gateable) interconnects, we speculate that Vdd-gating is able to reduce more energy than pure Vdd-selection does and have not studied this architecture class using Vdd-selection without Vdd-gating. Our speculation may be verified by future study.

The state-of-art commercial FPGAs have applied unidirectional routing switch in routing architecture and used depopulated local interconnects inside logic blocks [13], [26]. As these interconnect features may have a great impact on power and performance, in the future we will conduct architecture evaluation considering these features with Vdd programmability.

Recently reported work on FPGA energy reduction includes applying linear programming to allocate time slack to each routing tree sink and maximize energy reduction for Vdd-programmable interconnects without using Vdd-level converters (Class3 in this paper) [27], and performing co-optimization of device (Vdd & Vt) and FPGA architecture [28]. Applying

Heterogeneous Vt to logic blocks and interconnects, and architecture tuning can reduce ED product by 50% with no area overhead and no power-gating, and by 77% with power-gating.

ACKNOWLEDGEMENT

The authors like to thank Mr. Lerong Cheng and Ms. Ho-Yan Phoebe Wong for generating circuit models for a variety of basic FPGA circuits and for helpful discussions.

REFERENCES

[1] E. Kusse and J. Rabaey, "Low-energy embedded FPGA structures," in *Proc. Intl. Symp. Low Power Electronics and Design*, August 1998, pp. 155–160.
[2] 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.
[3] 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.
[4] T. Tuan and B. Lai, "Leakage power analysis of a 90nm FPGA," in *Proc. IEEE Custom Integrated Circuits Conf.*, 2003.
[5] J. H. Anderson, F. N. Najm, and T. Tuan, "Active leakage power optimization for FPGAs," in *Proc. ACM Intl. Symp. Field-Programmable Gate Arrays*, Februray 2004.
[6] J. Lamoureux and S. J. Wilton, "On the interaction between power-aware FPGA CAD algorithms," in *Proc. Intl. Conf. Computer-Aided Design*, November 2003, pp. 701–708.
[7] 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.
[8] 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.
[9] F. Li, Y. Lin, and L. He, "FPGA power reduction using configurable dual-vdd," in *Proc. Design Automation Conf.*, June 2004.
[10] 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*, Feb 2000, pp. 3–12.
[11] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers, Feb 1999.
[12] G. G. Lemieux and S. D. Brown, "A detailed router for allocating wire segments in field-programmable gate arrays," in *Proceedings of the ACM Physical Design Workshop*, April 1993.
[13] D. L. et al, "The stratix routing and logic architecture," in *Proc. ACM Intl. Symp. Field-Programmable Gate Arrays*, Feb 2003.
[14] F. Li and L. He, "Power modeling and characteristics of field programmable gate arrays," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 2004.
[15] U. of Berkeley Device Group, "Predictive technology model," in *http://www.device.eecs.berkeley.edu/ ptm/mosfet.html*, 2002.
[16] International Technology Roadmap for Semiconductor, in *http://public.itrs.net/*, 2002.
[17] Xilinx Corporation, "Virtex-II 1.5v platform FPGA complete data sheet," July 2002.
[18] D. E. Lackey and et al., "Managing power and performance for system-on-chip designs using voltage islands," in *Proc. Intl. Conf. Computer-Aided Design*, 2002, pp. 195 – 202.
[19] K. Usami and M. Horowitz, "Clustered voltage scaling techniques for low-power design," in *Proc. Intl. Symp. Low Power Electronics and Design*, 1995.
[20] C. Chen, A. Srivastava, and M. Sarrafzadeh, "On gate level power optimization using dual-supply voltages," *IEEE Trans. on VLSI Systems*, vol. 9, pp. 616–629, Oct 2001.
[21] M. Hamada, Y. Ootaguro, and T. Kuroda, "Utilizing surplus timing for power reduction," in *Proc. IEEE Custom Integrated Circuits Conf.*, 2001, pp. 89–92.
[22] K. Usami and et al, "Automated low-power technique exploiting multiple supply volgates applied to a media processor," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 3, pp. 463–472, 1998.
[23] M. Hamada and et al, "A top-down low power design technique using clustered voltage scaling with variable supply-voltage scheme," in *Proc. IEEE Custom Integrated Circuits Conf.*, 1998, pp. 495–498.
[24] R. Puri, L. Stok, J. Cohn, D. Kung, D. Pan, D. Sylvester, A. Srivastava, and S. Kulkarni, "Pushing ASIC performance in a power envelope," in *Proc. Design Automation Conf.*, 2003, pp. 788 – 793.
[25] Benton Calhoun and Frank Honore and Anantha Chandrakasan, "Design methodology for fine-grained leakage control in MTCMOS," in *Proc. Intl. Symp. Low Power Electronics and Design*, August 2003.
[26] D. L. et al, "The stratix ii routing and logic architecture," in *Proc. ACM Intl. Symp. Field-Programmable Gate Arrays*, Feb 2005.
[27] Y. Lin and L. He, "Leakage efficient chip-level dual-vdd assignment with time slack allocation for FPGA power reduction," in *Proc. Design Automation Conf.*, June 2005.
[28] L. C. et al, "Device and architecture co-optimization for FPGA power reduction," in *Proc. Design Automation Conf.*, June 2005.