Off-path Leakage Power Aware Routing for SRAM-based FPGAs

Keheng Huang[1,2], Yu Hu[1], and Xiaowei Li[1]
[1.]State Key Laboratory of Computer Architecture,
Institute of Computing Technology, Chinese Academy of Sciences
[2.]Graduate University of Chinese Academy of Sciences
{huangkeheng, huyu, lxw}@ict.ac.cn

Bo Liu[3,4], Hongjin Liu[3,4], Jian Gong[3,4]
[3.]Beijing Institute of Control Engineering,
[4.]Science and Technology on Space Intelligent Control Laboratory
liub@bice.org.cn, {lhjbuaa, gjxjtu1997}@hotmail.com

***Abstract*—As the feature size and threshold voltage reduce, leakage power dissipation becomes an important concern in SRAM-based FPGAs. This work focuses on reducing the leakage power in routing resources, and more specifically, the leakage power dissipated in the used part of FPGA device, which is known as the active leakage power. We observe that the leakage power in off-path transistors takes up most of the active leakage power in multiplexers that control routing, and strongly depends on Hamming distance between the state of the on-path input and the states of the off-path inputs. Hence, an off-path leakage power aware routing algorithm is proposed to minimize Hamming distance between the state of on-path input and the states of off-path inputs for each multiplexer. Experimental results on MCNC benchmark circuits show that, compared with the baseline VPR technique, the proposed off-path leakage aware routing algorithm can reduce active leakage power in routing resources by 16.79%, and the increment of critical-path delay is only 1.06%.**

***Index Terms*—leakage power, Hamming distance, off-path, routing, FPGAs.**

I. Introduction

Field programmable gate arrays (FPGAs) provide an attractive design platform due to their short design cycle and low development cost. With exponential growth in performance and capacity, SRAM-based FPGAs are widely used in many application domains such as telecommunication, industrial control, and embedded applications.

Although SRAM-based FPGAs provide all these advantages, as the feature size and threshold voltage (V_{th}) reduce, the overall leakage power is rapidly increasing [1][2]. Leakage power has two main components in modern integrated circuits: 1) subthreshold leakage; and 2) gate leakage. Subthreshold leakage is due to a nonzero current between the source and drain terminals of an off-state CMOS transistor. To mitigate performance degradation caused by a low supply voltage, threshold voltage is also reduced, leading to an exponential increase in subthreshold leakage. On the other hand, gate leakage is due to tunneling current through the gate oxide of a transistor. It increases exponentially with oxide thinning that aims to improve transistor's drive strength. Recent studies show that, in modern FPGAs, almost 60%-70% of the leakage power is dissipated in routing resources [3][4]. Consequently, the leakage problem in interconnect fabric is of great importance.

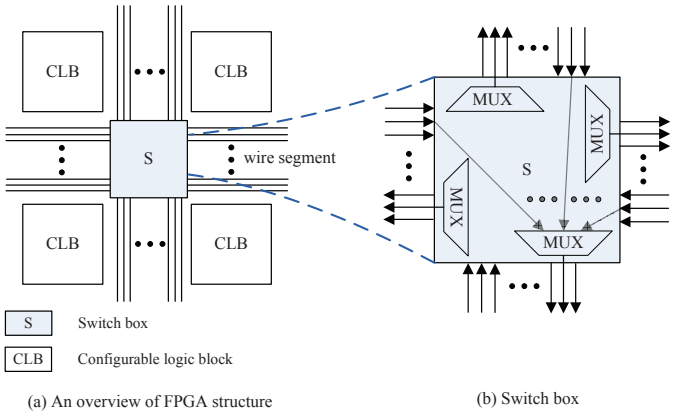


Fig. 1. Architecture of SRAM-based FPGAs.

For a design specification, the circuit is implemented in FPGA by storing the corresponding configuration bits into the pre-fabricated device. In general, an FPGA design uses only a portion of physical resources in the device. However, leakage power is dissipated in both the unused part of the device, namely *sleep leakage power*, and the used part of the device, namely *active leakage power*. For a typical design that occupies about 60% of the FPGA resources, active leakage power is more than sleep leakage power. For leakage power optimization, sleep leakage power is mainly reduced by putting the unused part into a sleep state, in which leakage is minimized [5][6][7], while active leakage power is optimized by multi-V_{th} based design and configuration bits inversion [8][9][10][11].

In this paper, we focus on active leakage power optimization in routing resources. First, we show the leakage power in off-path transistors takes up most of the dissipated leakage power. Then, we observe the off-path leakage power dissipation strongly depends on Hamming distance between the on-path input state and the off-path input states, and further characterize the relationship between them. Finally, based on the observation described above, an off-path leakage power aware routing algorithm is proposed to minimize off-path leakage power during the routing stage.

The rest of this paper is organized as follows. Section 2 introduces the background and previous work. Section 3 describes the leakage in multiplexer. Section 4 presents the motivation of our work. Section 5 introduces the procedure of our off-path

This work is supported in part by National Natural Science Foundation of China (NSFC Program) under Grant Nos. 61076018, 60803031, and 60921002, and in part by National Basic Research Program of China (973 Program) under Grant No. 2011CB302503.

978-3-9810801-8-6/DATE12/©2012 EDAA

leakage power aware routing algorithm. Experimental results are analyzed in Section 6. The conclusion and future work are given in Section 7.

II. Background and Previous work

A. Architecture of SRAM-based FPGAs

SRAM-based FPGAs have fixed number of configurable logic blocks (CLBs), switch boxes and wire segments, as Fig. 1 (a) shows. The CLB is a multi-input, multi-output digital circuits that is composed of many look-up tables (LUTs), multiplexers, and flip-flops (FFs). Alternatively, the switch box consists of a large number of multiplexers driving wire segments of different lengths to implement connections among CLBs, as Fig. 1 (b) shows.

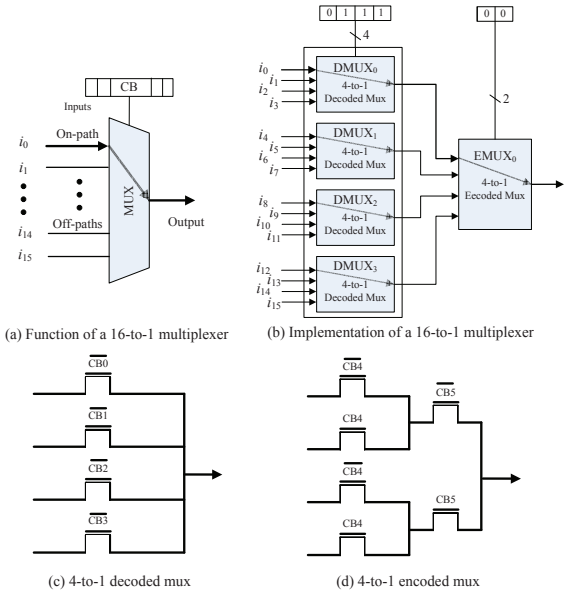
B. Previous work

There are many works focusing on leakage power optimization in FPGAs. The work proposed in [5][6][7] apply programmable supply voltage (V_{dd}) in FPGAs. The high supply voltage is applied to devices on critical paths to maintain performance, while the low supply voltage is applied to devices on noncritical paths to tradeoff performance and power, and zero voltage is applied to the sleep part of the device to reduce power. The work proposed in [8][9][10] use high threshold transistors in the device. Since leakage exponentially depends on the threshold voltage, the use of high threshold transistors can significantly reduce the leakage power of FPGAs. However, both the programmable supply voltage schemes and the high threshold transistor schemes may affect timing performance of the design. Furthermore, these schemes require architecture modifications, which may introduce additional area overhead.

There are also some leakage optimization techniques that are architecture independent. The work proposed in [11] observes the multiplexer with a logic 1 at the output has lower leakage power than that with a logic 0 at the output on average. Hence, the author inverts the configuration bits in LUTs to obtain more logic 1s in multiplexers, resulting in lower active leakage power. Note that, the work in [11] only takes the average leakage power into consideration. However, as will be described in Section III, the leakage power mainly depends on the relationship of the logic states between the on-path input and off-path inputs, which motivates us to explore how the relationship can be used for leakage power optimization.

III. Leakage in multiplexer

Since multiplexers are the key component of switch box, and take up most of the leakage power dissipation, a brief analysis of leakage power in multiplexer is performed first. The multiplexers range from small 4-input multiplexers to large 36-input multiplexers. Fig. 2 (a) gives an example of a 16-to-1 multiplexer. When the multiplexer is used in a design, only one input is selected from the sixteen inputs to the output. Specifically, the path is called *on-path*, the input is called *on-path input*, and the transistors along on-path is called *on-path transistors*. On the other hand, other fifteen inputs are called *off-paths*, the inputs are called *off-path inputs*, and the transistors along off-paths are called *off-path transistors*.



(a) Function of a 16-to-1 multiplexer (b) Implementation of a 16-to-1 multiplexer

(c) 4-to-1 decoded mux (d) 4-to-1 encoded mux

Fig. 2. Detailed architecture of multiplexer.

An implementation of an NMOS transistor based 16-to-1 multiplexer is shown in Fig. 2 (b) [12]. The multiplexer is composed of two levels. At the first level, there are four 4-to-1 decoded multiplexers (DMUX), as Fig. 2 (c) shows. The four configuration bits at this level select one of four inputs from each multiplexer's inputs to multiplexer's output. At the second level, there is a 4-to-1 encoded multiplexer (EMUX), which further selects one input from the former selected four inputs at the first level to the final output, as Fig. 2 (d) shows. In this example, i_0, i_4, i_8, and i_{12} are selected at the first level, and i_0 is further selected at the second level. Therefore, i_0 is the on-path input, and transistors along the path are on-path transistors. On the other hand, $i_1 \sim i_{15}$ are off-path inputs, and transistors along the paths are off-path transistors.

According to [4][11], the leakage power consumption of a multiplexer strongly depends on the logic value of input vectors. In this work, we further observe that, most of the leakage power in a multiplexer is consumed by off-path transistors. The same as the work in [11], we take $DMUX_0$ in Fig. 2 (b) as an example. SPICE simulations (at 110°C) on PTM 65nm NMOS models [13] were performed to measure the leakage power of the multiplexer. As Fig. 3 shows, 80.59% of the leakage power is consumed by off-path transistors on average. Note that, this percentage will further grow by additionally considering other off-path transistors in Fig. 2 (b). Consequently, reducing off-path leakage power is of significant importance.

Furthermore, we observe that the off-path leakage power actually depends on the relationship of logic states among the on-path input and off-path inputs. Also take $DMUX_0$ as an example. If the logic state of on-path input i_0 is 0, the off-path leakage power strongly depends on the number of 1s at the off-path inputs, and vice versa, as Fig. 4 shows. In the extreme case, if all off-path inputs have the same logic state as the on-path input, such as 0000 and 1111, the off-path leakage power is minimized. That is because there is no voltage difference between the source and drain of off-path transistors, the leakage power dissipation is low. In contrast, if all off-path inputs have

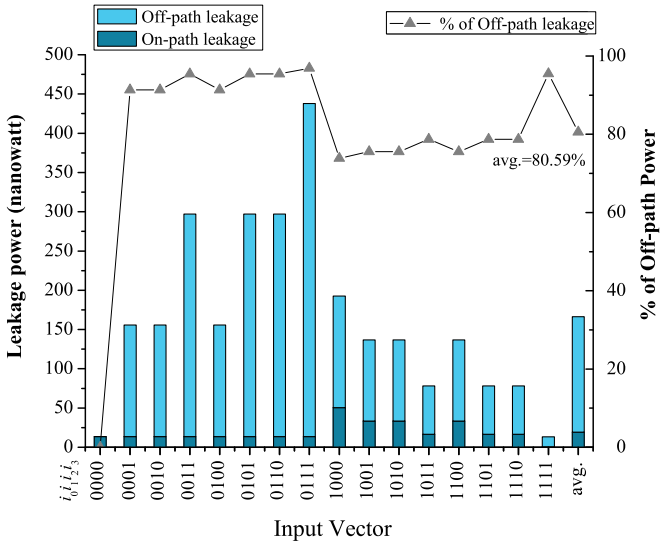


Fig. 3. Percentage of off-path leakage power.

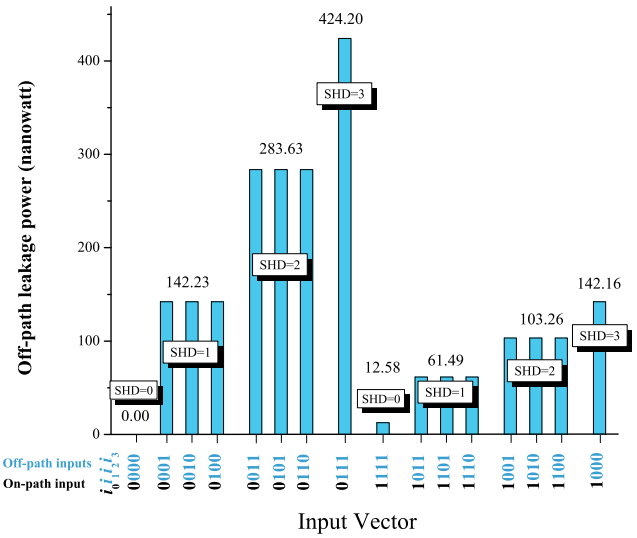


Fig. 4. The relationship between SHD and off-path leakage power.

the reverse logic state to the on-path input, such as 0111, and 1000, the off-path leakage power is maximized. That is because a voltage difference exists between source and drain of each off-path transistor, which results in huge leakage power dissipation.

We use a metric namely *State Hamming Distance* (SHD) to accurately and efficiently describe the relationship of logic states among on-path input and off-path inputs. For input vector v, SHD describes the sum of hamming distances between the logic state of on-path input , denoted as $State_{on-path}$, and the logic state of each off-path input i, denoted as $State^{i}_{off-path}$, which can be computed as follows:

$$SHD_v = \sum_{i=0}^{N_{off-path}} H(State_{on-path}, State^{i}_{off-path}) \quad (1)$$

where $N_{off-path}$ represents the total number of off-paths.

Take input vector 0111 as an example, the state of the on-path input is 0, and the states of off-path inputs are 1s, then the SHD for input vector 0111 is computed as follows:

$$\begin{aligned} SHD_{0111} &= H(0,1) + H(0,1) + H(0,1) \\ &= 1 + 1 + 1 \\ &= 3 \end{aligned}$$

In this way, the leakage power of a multiplexer is optimized if we minimize the average SHD for all possible input vectors of the multiplexer, as Fig. 4 shows.

IV. Motivation

Since the connections among multiplexers are implemented during the routing stage, the SHD of each multiplexer is determined during the routing stage also, which motivates us to perform off-path leakage aware routing for SRAM-based FPGAs. Note that, the logic state of each wire in a design is not a constant value, but with a probability to be logic 1, namely *logic state probability*, denoted as $P(1)$. In the worst case, if the logic state probability of each wire is close to 0.5, which indicates any input vector has a similar probability occurring at the inputs of multiplexer, there are not any alternative routing choices for SHD reduction, and no space for leakage power optimization. Hence, the optimization space of our work is affected by how far the logic state probability of each wire is away from 0.5, namely *probability skew*. For example, if the logic state probability of all wires in a design is either 0.2 or 0.8, the probability skew is 0.3 (i.e. 0.5-0.2=0.3 or 0.8-0.5=0.3) for this circuit, as Fig. 5 shows. We conducted experiments on twenty MCNC benchmark circuits to estimate the logic state probability of each wire by ACE tool [14], and collected the probability skew for each circuit. As Fig. 6 shows, the average probability skew for all benchmark circuits is 0.37, which indicates a large space for leakage power optimization by our work.

V. Our Approach

In this work, with the consideration of SHD for each multiplexer, an off-path leakage power aware routing for FPGAs is proposed. The algorithm is a three-step procedure. First, a leakage aware routing graph is abstracted from the implemented design at each routing iteration. Afterwards, the leakage power estimation for the design is introduced. Finally, an off-path leakage power aware router is proposed to currently optimize timing, congestion, and leakage power.

A. Establishing leakage aware routing graph

In FPGAs, the connections among CLBs are implemented by switch boxes and wire segments. A leakage aware routing graph is constructed by abstracting the leakage power information of the design. Fig. 7 depicts an example of leakage aware routing graph that implements $wire_1$ from wire segment S_1 to S_2, and $wire_2$ from S_3 to S_4. Due to the flexibility of FPGAs, there are also some other free routing resources that are not used in the device, such as S_5, S_6, S_7, S_8, and the multiplexers among them. Note that, an output wire segment has more than one possible input, so the multiplexer may connect one of the input wire segments to the output. In leakage aware routing graph, the on-paths are represented by bold lines, and the off-paths are represented by dashed lines. The logic state probability for each wire segment is computed after logic simulation on the netlist. Hence, the occurrence probability of each SHD state for the multiplexer can be calculated.

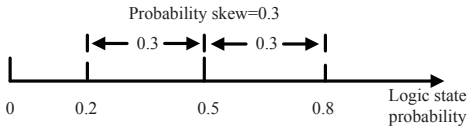


Fig. 5. Example of probability skew.

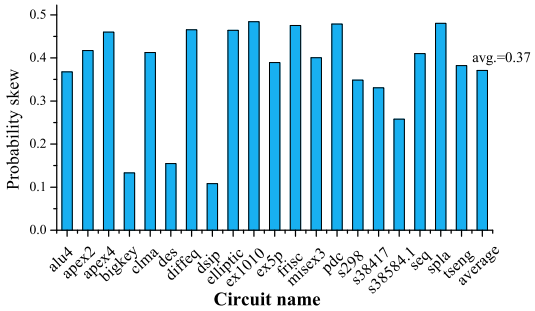


Fig. 6. Probability skew for all benchmark circuits.

By abstracting the leakage aware routing graph, the interconnections among wire segments are clearly described. In addition, based on the occurrence probability of each SHD state and the leakage power of the corresponding SHD state, the leakage aware routing graph can be used to estimate the leakage power of the design, and be further used to guide leakage aware routing stage.

B. Leakage power estimation

Based on the leakage aware routing graph, leakage power estimation is performed for the design at each routing iteration. First, the leakage power for a multiplexer is the weighted summation of the leakage power dissipated for all SHD states, which can be computed as follows:

$$L_{mux} = \sum_{v=0}^{N_{SHD}} P(SHD_v) * L(SHD_v) \quad (2)$$

where N_{SHD} is the total number of SHD states for the multiplexer. $P(SHD_v)$ represents the occurrence probability of state SHD_v, and $L(SHD_v)$ represents the corresponding leakage power dissipated in the state SHD_v.

For $P(SHD_v)$ in Equation (2), the occurrence probability of the state SHD_v is the product of probability that the logic of each input i satisfies the logic of the i^{th} bit in the input vector v, which is given by:

$$P(SHD_v) = \prod_{b_i \in v} P(b_i) \quad (3)$$

where b_i is the i^{th} bit of the input vector v. $P(b_i)$ represents the probability that the logic of the input i is b_i. If b_i is 1, $P(b_i)$ equals to the logic state probability $P(1)$ of the input i. On the other hand, if b_i is 0, $P(b_i)$ equals to $1 - P(1)$ of the input i. The logic state probability of each wire segment is stored in the leakage aware routing graph.

For $L(SHD_v)$ in Equation (2), SPICE simulation based on the PTM model is performed to obtain the leakage power for each corresponding SHD state, as is shown in Fig. 4. Based on the two parts described above, we can estimate the leakage power of the multiplexer.

Take the design shown in Fig. 8 (a) as an example, which shows the connections among S_1, S_2, and S_3 shown in Fig. 7. The logic state probabilities of input S_1 and S_3 are

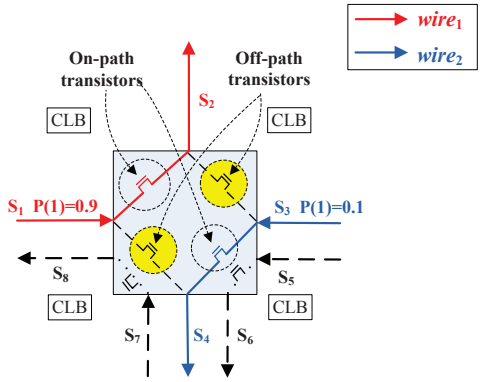
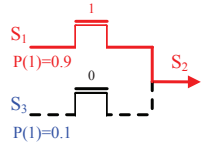


Fig. 7. Example of a leakage aware routing graph.



(a) Example of a 2-input multiplexer

SHDv	L_{SHDv}
SHD_{00}	1
SHD_{11}	3
SHD_{01}	140
SHD_{10}	80

(b) Leakage power for the corresponding SHD state

Fig. 8. Example of leakage power estimation.

$P_{S1}(1) = 0.9$, $P_{S3}(1) = 0.1$, respectively. Then, the occurrence probability of SHD_{00} for the multiplexer is

$$\begin{aligned} P(SHD_{00}) &= (1 - P_{S1}(1)) * (1 - P_{S3}(1)) \\ &= (1 - 0.90) * (1 - 0.10) \\ &= 0.09 \end{aligned}$$

The occurrence probabilities of SHD_{01}, SHD_{10}, and SHD_{11} are estimated in the similar way. Afterwards, the leakage power dissipated for the corresponding SHD state is obtained through SPICE simulation, as Fig. 8 (b) shows. Finally, the leakage power of the multiplexer is

$$\begin{aligned} L_{mux} &= P(SHD_{00}) * L(SHD_{00}) + P(SHD_{01}) * L(SHD_{01}) \\ &\quad + P(SHD_{10}) * L(SHD_{10}) + P(SHD_{11}) * L(SHD_{11}) \\ &= 0.09 + 1.4 + 64.8 + 0.27 \\ &= 66.56 \end{aligned}$$

Afterwards, when implementing a wire segment S_n, the leakage power dissipation may occur at two multiplexers. The first one is the multiplexer that connects wire segments to the source of S_n. The leakage power dissipated here is represented as $L_{mux}(source - S_n)$. The second is the multiplexer that connects the sink of S_n to other wire segments. The leakage power dissipated here is represented as $L_{mux}(sink - S_n)$. The leakage power $L(S_n)$ that occurs when the implementation choose S_n is as follows:

$$L(S_n) = L_{mux}(source - S_n) + L_{mux}(sink - S_n) \quad (4)$$

where both $L_{mux}(source - S_n)$ and $L_{mux}(sink - S_n)$ can be estimated by Equation (2).

Finally, by traversing all multiplexers in the leakage aware routing graph, the leakage power of the whole design is the summation of leakage power dissipated in each multiplexer, which can be computed as follows:

$$L_{design} = \sum_{i=0}^{N_{mux}} L_{muxi} \quad (5)$$

where N_{mux} is the total number of multiplexers in the design.

TABLE I
COMPARISON OF ACTIVE LEAKAGE POWER, TIMING PERFORMANCE AND TOTAL WIRE LENGTH

Circuit characteristics				Active leakage power(nanowatt)			Critical-path delay(s)			Total wire length(# of CLBs spanned)		
Circuit	PI#	PO#	reg#	VPR(baseline)	Leakage aware	Ratio	VPR(baseline)	Leakage aware	Ratio	VPR(baseline)	Leakage aware	Ratio
alu4	14	8	0	8.08E+05	6.84E+05	84.67%	1.17E-08	1.14E-08	97.68%	19563	19964	102.05%
apex4	9	19	0	1.45E+06	1.21E+06	83.40%	1.14E-08	1.11E-08	97.79%	30900	31862	103.11%
misex3	14	14	0	7.25E+05	6.18E+05	85.18%	1.05E-08	1.02E-08	97.02%	18259	18957	103.82%
pdc	16	40	0	4.17E+06	3.61E+06	86.67%	1.05E-08	1.14E-08	108.29%	79294	80144	101.07%
s298	4	6	14	7.22E+03	4.50E+03	62.24%	1.86E-09	2.14E-09	115.36%	367	417	113.62%
spla	16	46	0	4.56E+06	3.78E+06	82.87%	1.75E-08	1.81E-08	103.43%	87958	89652	101.93%
ex1010	10	10	0	1.56E+06	1.29E+06	82.51%	1.17E-08	1.13E-08	96.74%	37558	37794	100.63%
ex5p	8	63	0	6.83E+05	6.18E+05	90.57%	7.88E-09	7.88E-09	99.97%	16920	17583	103.92%
apex2	38	3	0	1.57E+06	1.32E+06	83.98%	1.17E-08	1.20E-08	102.15%	34129	36117	105.82%
bigkey	229	197	224	7.24E+05	5.87E+05	81.13%	7.47E-09	7.46E-09	99.92%	39308	38503	97.95%
clma	62	82	33	6.91E+06	6.23E+06	90.21%	1.11E-08	1.24E-08	112.10%	152210	161003	105.78%
des	256	245	0	8.87E+05	7.29E+05	82.14%	1.03E-08	8.86E-09	85.98%	42540	42255	99.33%
diffeq	28	3	305	7.57E+05	6.38E+05	84.34%	5.74E-09	5.76E-09	100.36%	20918	22329	106.75%
dsip	229	197	224	9.62E+05	7.77E+05	80.78%	7.62E-09	8.25E-09	108.26%	45254	45679	100.94%
elliptic	3	2	194	3.01E+05	2.51E+05	83.67%	6.39E-09	6.26E-09	98.03%	9529	9975	104.68%
seq	41	35	0	1.54E+06	1.33E+06	86.71%	9.13E-09	9.07E-09	99.35%	36502	38408	105.22%
tseng	52	122	382	4.97E+05	4.07E+05	81.92%	7.97E-09	7.88E-09	98.81%	23155	23174	100.08%
frisc	20	116	886	3.98E+06	3.34E+06	84.04%	1.35E-08	1.36E-08	100.53%	106646	104450	97.94%
s38417	29	106	1462	2.26E+06	1.87E+06	82.72%	1.03E-08	1.03E-08	99.72%	71755	75508	105.23%
s38584.1	38	304	1260	3.11E+06	2.62E+06	84.38%	9.65E-09	9.62E-09	99.69%	118110	113103	95.76%
Geomean				1.87E+06	1.60E+06	**83.21%**	9.70E-09	9.75E-09	**101.06%**	49544	50343	**102.78%**

C. Off-path leakage aware router

Conventional FPGA router uses a maze-based heuristic algorithm that considers timing and congestion. The cost function of VPR router to evaluate a wire segment S_n while forming a connection from source i and sink j is as follow:

$$Cost(S_n) = Crit(i,j) * delay(S_n) + (1 - Crit(i,j)) * b(S_n) * h(S_n) * p(S_n) \quad (6)$$

where $Cost(S_n)$ is composed of a timing term and a congestion term. In the timing term, $Crit(i,j)$ is an indication of the connection (i,j) is how close to the critical path, and $delay(S_n)$ is the timing delay of S_n. In the congestion term, $b(S_n)$ is the base cost, $h(S_n)$ is the historical congestion cost, and $p(S_n)$ is the present congestion cost of S_n.

For the off-path leakage aware router, the main challenge is the integration of the leakage cost into a conventional congestion-delay algorithm. When choosing a wire segment S_n from a set of candidates, the leakage cost $L(S_n)$ is added into the cost function of the router as follows:

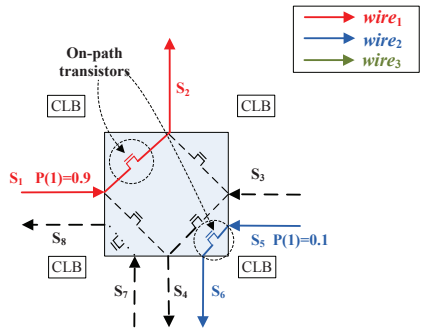
$$Cost(S_n) = Crit(i,j) * delay(S_n) + (1 - Crit(i,j)) * b(S_n) * h(S_n) * p(S_n) * L(S_n) \quad (7)$$

Based on the cost function described in Equation (5), the router will concurrently consider timing, congestion, and leakage power during routing stage. Also take the connection in Fig. 7 as an example. Suppose $wire_1$ has been routed, then the problem is how to route $wire_2$ to make the utilized multiplexers have the minimum leakage power. If there are free wire segments that can be used to route $wire_2$, the router will choose the ones that have no off-path transistors connecting to the already routed wires, e.g. choose S_5 and S_6 which have no off-path transistors connecting to $wire_1$, as shown in Fig. 9 (a). In this case, the SHDs of the multiplexers are minimized. On the other hand, if there are no alternative free wire segments, the router will swap the routed wires (e.g. $wire_2$ and $wire_3$) so that the two wires (e.g. $wire_1$ and $wire_3$) which could be possibly connected by off-path transistors that have approximate logic state probabilities (e.g. $P_{wire1}(1) = 0.9$, $P_{wire3}(1) = 0.8$), as shown in Fig. 9 (b). As a result, the average SHD for all input vectors of each multiplexer is reduced, which results in lower leakage power dissipation of the design.

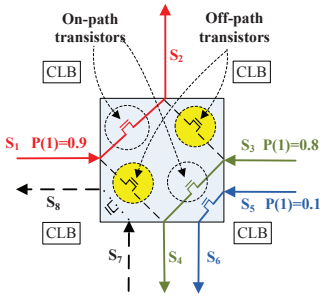
According to Equation (5), the leakage power of original routing shown in Fig. 7 is 181.12nw. In contrast, the leakage power of the optimized routing results shown in Fig. 9 (a) and (b) are 4.0nw and 62.76nw, respectively, which are much lower than that of the original routing results.

VI. EXPERIMENTAL RESULTS

In order to validate the proposed off-path leakage power aware routing technique, we use the twenty largest MCNC benchmark circuits, and map them into LUTs by Berkeley ABC mapper [15]. During the stage of placement and routing, VPR



(a) Optimized routing result by using free wire segments



(b) Optimized routing result by swapping routed wires

Fig. 9. Example of optimized routing results.

TABLE II
COMPARISON OF DYNAMIC POWER

Circuit	Dynamic power(nanowatt)		Ratio
	VPR(baseline)	Leakage aware	
alu4	0.56	0.60	106.20%
apex4	0.59	0.64	108.43%
misex3	0.47	0.53	112.22%
pdc	0.76	0.80	105.57%
s298	0.008	0.009	116.04%
spla	0.10	0.11	106.00%
ex1010	0.50	0.51	103.04%
ex5p	0.36	0.38	106.54%
apex2	0.95	1.09	114.98%
bigkey	1.53	1.59	103.45%
clma	1.79	2.01	112.14%
des	1.58	1.66	105.13%
diffeq	0.21	0.22	103.00%
dsip	2.26	2.42	107.16%
elliptic	0.12	0.14	111.23%
seq	0.91	1.02	111.85%
tseng	0.41	0.45	109.43%
frisc	0.87	0.89	102.68%
s38417	1.94	2.23	114.79%
s38584.1	3.91	3.90	99.58%
Geomean	1.04	1.11	**107.97%**

(Versatile Place and Route) [16], the state-of-the-art placement and routing tool for FPGAs, is used. The architecture used for all experiments is a Virtex-like FPGA architecture. Each CLB contains four 6-input LUTs. The route channel width is set as 200, and contains four types of wire segments, i.e. single(8%), double(20%), hex(60%), and long(12%). All the proposed algorithms are implemented in Java and run on a Xeon Linux workstation with 6GB memory.

We have compared two CAD flows in our experiments. One is the baseline which uses the original VPR toolset to implement a mapped netlist. The other is the proposed flow, where the off-path leakage aware routing is used after placement.

The comparison of active leakage power, timing performance and routing congestion are shown in Table I. Compared with the baseline result of VPR router (labeled as *VPR* in Table I), the proposed off-path leakage-aware router (labeled as *Leakage aware* in Table I) can reduce active leakage power by 16.79% on average. In the aspect of timing performance, the critical-path delay increase is 1.06% across all benchmark circuits. Finally, we use the total wire length to describe the routing congestion of a design. As shown in Table I, the total wire length increase of the proposed scheme is 2.78% on average. Hence, the proposed off-path leakage aware routing scheme can significantly reduce active leakage power, while has little effect on timing performance and routing congestion of the design.

On the other hand, the proposed off-path leakage aware routing algorithm modifies the implementation of the design, which may also affect the dynamic power of the design. Referring to [17], the comparison of dynamic power consumption between the conventional VPR and the proposed scheme are given in Table II. Experimental results show that, compared with the results of VPR, the dynamic power consumption of the proposed scheme is increased by 7.97%. Note that, the average dynamic power across twenty MCNC benchmark is only 1.11nw, which is six to seven orders of magnitude lower than the active leakage power consumption. Taking the scale of benchmark circuits into consideration, the dynamic power consumption is also so small. Hence, the increase is not that significant. The similar dynamic power results are given in [18], and the similar leakage power results are given in [11] also.

VII. CONCLUSION AND FUTURE WORK

In this work, we observe that the off-path leakage power takes up most of the active leakage power, and strongly depends on the SHD of the multiplexer. By importing the SHD aware leakage power cost into the original VPR router, the proposed off-path leakage aware routing algorithm can reduce the active leakage power in routing resources by 16.79%, and the critical-path delay is increased by only 1.06%. In the future, we will further evaluate the effectiveness of the proposed scheme in various FPGA architectures and in various routing congestion scenarios.

REFERENCES

[1] Deming Chen, Jason Cong, and Peichen Pan, "FPGA Design Automation: A survey," Foundations and Trends in Electronic Design Automation, Vol. 1, No. 3, November 2006.

[2] The International Technology Roadmap for Semiconductors (ITRS), 2009 edition, in http://public.itrs.net.

[3] A. Rahman, and V. Polavarapuv, "Evaluation of Low-leakage Design Techniques for Field-programmable Gate Arrays," in Proc. of ACM/SIGDA Int. Symp. Field Programmable Gate Arrays (FPGA), 2004, pp. 23-30.

[4] T. Tuan, B. Lai, "Leakage Power Analysis of a 90nm FPGA," in Proc. of IEEE Custom Integrated Circuits Conf. (CICC), 2003, pp. 57-60.

[5] Fei Li, Yan Lin, and Lei He, "Field Programmability of Supply Voltages for FPGA Power Reduction," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 26, No. 4, April 2007, pp. 752-764.

[6] M. Anis, S. Areibi, M. Mahmoud, and M. Elmasry, "Dynamic and leakage power reduction in MTCMOS circuits using an automated efficient gate clustering technique," in Proc. of IEEE/ACM Design Automation Conf. (DAC), 2002, pp. 480-485.

[7] T. Sakurai, "Minimizing power across multiple technology and design levels," in Proc. of IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD), 2002, pp. 24-27.

[8] Liqiong Wei, Zhanping Cheng, and et al, "Design and Optimization of Low Voltage High Performance Dual Threshold CMOS Circuits," in Proc. of IEEE/ACM Design Automation Conf. (DAC), 1998, pp. 489-494.

[9] C. Kim and K. Roy, "Dynamic V_{th} Scaling Scheme for Active Leakage Power Reduction," in Proc. of IEEE/ACM Design, Automation and Test in Europe Conf. (DATE), 2002, pp. 163-167.

[10] L. Ciccarelli, A. Lodi, and et al, "Low leakage circuit design for FPGAs," in Proc. of IEEE Custom Integrated Circuits Conf. (CICC), 2004, pp. 715-718.

[11] Jason H. Anderson, Farid N. Najm, "Active Leakage Power Optimization for FPGAs," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 25, No. 3, March 2006, pp. 423-437.

[12] G. Lemieux and D. Lewis, "Circuit design of routing switches," in Proc. of ACM/SIGDA Int. Symp. Field Programmable Gate Arrays (FPGA), 2002, pp. 19-28.

[13] Y. Cao, T. Sato, and et al, "New Paradigm of Predictive MOSFET and Interconnect Modeling for Early Circuit Simulation," in Proc. of IEEE Custom Integrated Circuits Conf. (CICC), 2000, pp. 201-204.

[14] N. C. K. Choy, S. Chin, and et al, "Power Modeling for FPGAs." in http://www.ece.ubc.ca/ stevew/powermodel.html

[15] "ABC: A system for sequential synthesis and verification," in http://www.eecs.berkeley.edu/~alanmi/abc/.

[16] V. Betz and J. Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," in Proc. of Int. Workshop on Field-Programmable Logic and Applications. 1997, pp. 213-222.

[17] K. Poon, S. Wilton, and A. Yan, "A Detailed Power Model for Field-Programmable Gate Arrays," ACM Transactions on Design Automation of Electronic Systems, Vol. 10, No. 2, April 2005, pp. 279-302.

[18] Julien Lamoureux, Steven J.E Wilton, "On The Interaction Between Power-aware FPGA CAD Algorithms," in Proc. of IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD), 2003, pp. 701-708.