An Analytical State Dependent Leakage Power Model for FPGAs

Akhilesh Kumar and Mohab Anis
Department of Electrical and Computer Engineering
University of Waterloo, Waterloo, ON, Canada N2L3G1
{a5kumar, manis}@vlsi.uwaterloo.ca

Abstract

In this paper we present a state dependent analytical leakage power model for FPGAs. The model accounts for subthreshold leakage and gate leakage in FPGAs, since these are the two dominant components of total leakage power. The power model takes into account the dependency of gate and subthreshold leakage on the probability of the state of circuit inputs. The leakage power model has two main components, one which computes the probability of a state for a particular FPGA circuit element, and the other which computes the leakage of the FPGA circuit element for a given input using analytical equations. This FPGA power model is particularly important for rapidly analyzing various FPGA architectures across different technology nodes.

1 Introduction

The scaling of technology has led to an exponential increase in leakage power, and managing leakage power has emerged as a key design challenge. In earlier FPGA designs,leakage was not a concern, however, contemporary FPGAs are being implemented in sub 100nm CMOS technologies, and the leakage power cannot be ignored. The work in [2] showed that a 90nm FPGA consumes too much leakage power to be successfully used in mobile applications. Managing leakage power in FPGAs is therefore necessary for FPGAs to retain its competitive advantages over high performance custom VLSI designs and also for gaining popularity in domains such as wireless personal communications and low power biomedical applications. It is important, therefore, to accurately model the various components of leakage power and to analyze its behavior in FPGAs, so that leakage reduction techniques can be effective and efficient.

The work in [7] discussed various leakage current mechanisms and leakage reduction techniques for CMOS circuits. Analytical equations for leakage computation have been studied and developed in detail, which can model the complex behavior of various components of leakage current in a MOS transistor. These models are based on physical and empirical parameters [8]. Typically, the leakage power consumption of any circuit is not only dependent on the physical parameters of the circuit, but is also heavily dependent on the inputs to the circuit. The work in [9] shows that the leakage current can vary by an order of magnitude depending on the input to the circuit and demonstrated that certain inputs are the dominant leakage states for a gate.

There have been very few works targeted at modeling leakage power for the FPGAs. The work in [5] modeled the dynamic and the leakage power. However, it considered only subthreshold leakage and did not consider the dependency of subthreshold leakage on the state of the circuit, rather it calculated an average leakage considering that all the transistors were leaking and the V_{gs} was considered for as half of V_{th} for leakage computation. This produces inaccurate estimation of leakage current. The work in [3] and [2] calculated total power using look-up table based approach using SPICE simulations to characterize the power of the FPGA circuit elements. However, it did not develop any state dependent leakage power model and the methodology described is technology dependent.

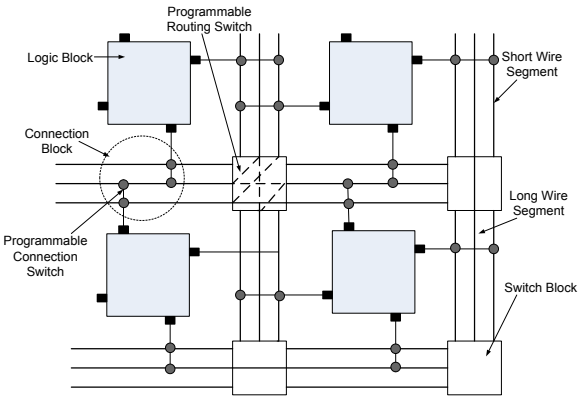

Figure 1: FPGA architecture under consideration

Motivated by the above mentioned limitations of the previous works, this work develops an analytical model for leakage power calculation for FPGAs, that takes into account the dependency of the leakage power on the state of the circuit. The contribution of the paper can be summarized as: (1) Developing analytical models and methodology to compute subthreshold and gate leakage power for FPGAs, independent of the technology node,(2) computation of state dependent subthreshold and gate leakage, and (3) analysis of sources of leakage in FPGAs.

2 Targeted FPGA Architecture

The FPGA architecture is very regular in structure. Fig. 1 shows the targeted FPGA architecture for this work. It has two main components - logic blocks (CLBs) and routing resources. The logic blocks implement the functionality of the given circuit while the routing resources provide the connectivity for implementing the

3-9810801-0-6/DATE06 © 2006 EDAA

logic. The most popular FPGA architecture is the SRAM based architecture which is described below and is used in this work [4]. The logic block of the SRAM based FPGA are composed of basic logic elements (BLE). Each BLE consists of a k-input LUT, flip-flop and a multiplexer for selecting the output either directly from the output of LUT or the registered output value of the LUT stored in the flip-flop. LUT is an array of SRAM cells. In the cluster based logic block, the logic block is made up of N BLEs. In this work, island-based routing architecture is used, in which the routing resources form a mesh like structure with the horizontal and vertical routing channels. These routing channels are connected by switch blocks which are programmable and thus provide the flexibility in making the connections. The logic blocks are connected to the routing channels through the connection blocks which are also programmable.

3 Leakage Modeling in FPGAs

3.1 Analytical Models for Leakage Currents

The leakage power models will consider the subthreshold and the gate leakage. The following are the subthreshold and gate leakage equations used in the power model [8].

$$I_{sub} = I_0\left[1 - exp\big[\frac{-V_{ds}}{V_T}\big]\right].exp\left[\frac{V_{gs} - V_{th} - V_{off}}{nV_T}\right] \quad (1)$$

$$I_0 = \mu\frac{W}{L}\sqrt{\frac{q\epsilon_{si}NDEP}{2\phi_s}V_T^2} \quad (2)$$

$$I_{gc0} = \frac{W.L.A.V_{gs}.V_{aux}}{T_{ox}^2}.exp\Big[-B.T_{ox}.\Big(AIGC - BIGC.V_{oxdepinv}\Big).\Big(1 + CIGC.V_{oxdepinv}\Big)\Big] \quad (3)$$

$$V_{aux} = NIGC.V_T.log\big[1 + exp\big(\frac{Vgs - V_{th0}}{NIGC.V_T}\big)\big] \quad (4)$$

$$V_{oxdepinv} = K1.\sqrt{\phi_s} + V_{gs} - V_{th} \quad (5)$$

$$I_{gcs} = \frac{PIGCD.V_{ds} + exp(-PIGCD.V_{ds})}{PIGCD^2.V_{ds}^2 + 2e - 4} - \frac{1 + 1e - 4}{PIGCD^2.V_{ds}^2 + 2e - 4} \quad (6)$$

$$I_{gcd} = \frac{1 - (PIGCD.V_{ds} + 1).exp(-PIGCD.V_{ds})}{PIGCD^2.V_{ds}^2 + 2e - 4} + \frac{1e - 4}{PIGCD^2.V_{ds}^2 + 2e - 4} \quad (7)$$

The subthreshold leakage (I_{sub}) equations [8] are given by equations (1) and (2), where V_T is the thermal voltage, V_{off} is the offset voltage which determines the channel current at $V_{gs} = 0$, n is the subthreshold swing coefficient, $W, L, \mu, q, \phi_s, \epsilon_{si}$, are the width, length, mobility of charge carriers, electron charge, surface potential and permittivity of silicon, respectively, for the transistor. Since only the gate to channel current (I_{gc0})is the dominant gate leakage current, and the gate current for the PMOS is significantly smaller that the gate current for the NMOS, we consider the modeling of gate to channel current only for the NMOS [10]. However, the proposed model can be easily extended to incorporate other gate leakage components and the gate leakage for the PMOS. The gate leakage equations are given by (3)- (7), where A, B are physical constants, T_{ox} is the gate oxide thickness, $AIGC$, $BIGC$, $CIGC$, and $NIGC$ are the empirical parameters, $K1$ is the first order body bias coefficient. Equation (3) is used for computing I_{gc0} and equations (6), and (7) are used for partitioning I_{gc0} into the source current I_{gcs} and drain current I_{gcd}, where $PIGCD$ is a parameter for the partitioning.

In this work, we used industrial CMOS 130nm and CMOS 90nm processes for the leakage analysis of the FPGA using our leakage power model. The deep-submicron MOSFETs have various short channel effects (SCE) which were not present in long channel devices. For the CMOS 130nm process that we used, we observed that the threshold voltage (V_{th}) of the NMOS was affected by the reverse narrow width effect (RNWE) [7], i.e., the threshold voltage of the transistor increased as the width of the transistor increased from the minimum width, which consequently reduced the leakage of the transistor. Further, the threshold voltage of the transistors are also affected by the drain to source voltage (V_{ds}). The threshold voltage of the transistor decreases when the drain to source voltage is increased. To incorporate these effects into our models, we fitted the experimental data from the SPICE simulation to empirical equations as follows:

$$V_{th}|_{(V_{ds}=0)} = V_0\big(1 - a.exp(-b_1.W - b_2.W^2)\big) \quad (8)$$

$$V_{th} = V_{th}|_{(V_{ds}=0)} - m.V_{ds} \quad (9)$$

where equation 8 models the RNWE, and equation 9 models the impact of V_{ds} on V_{th}. For CMOS 130nm NMOS, $V_0 = 0.412V$, $a = 0.345003$, $b_1 = 1.01194$, $b_2 = -0.0568004$, and $m = 0.02125$. For CMOS 130nm PMOS, the RNWE was not too significant so we modeled only the dependence of V_{th} on V_{ds}, with $m = 0.02388$. These values were determined from curve fitting of the actual simulation data. For the CMOS 90nm process we observed a similar RNWE for the NMOS, and the dependence of V_{th} on V_{ds} was observed for both NMOS and PMOS. However, for the CMOS 90nm PMOS we observed a narrow width effect (NWE)[7] which results in increasing V_{th} as the width of the transistor is decreased. The RNWE and V_{ds} dependence for CMOS 90nm NMOS were modeled using equations (8) and (9) with the constants as $V_0 = 0.320812$, $a = 0.437178$, $b_1 = 1.2$, $b_2 = -0.068$, and $m = 0.0668$. For the PMOS we developed a model using curve fitting to account for the NWE, as follows:

$$V_{th} = \frac{f_1 + f_2.W + f_3.W^2}{g_1 + g_2.W + g_3.W^2} \quad (10)$$

where $f_1 = 0.49$, $f_2 = 1.16679$, $f_3 = -1.51$, $g_1 = 0.318$, $g_2 = 4.3$ and $g_1 = -0.533$. The impact of V_{ds} was modeled using equation (9), with $m = -0.0468$. We have used these equations in our power model to model the reverse narrow width effect and the dependence of V_{th} on V_{ds}. Although, the constants used in these equations make them technology dependent, these data can be easily extracted by simulating only one device with few different widths and drain to source voltages.

Table 1 shows that the inclusion of the RNWE in the power model greatly improves the overall accuracy of the power model. We have determined the base threshold voltages of the devices from the SPICE simulation so that various effects can be accounted for in the model.

Table 1: Comparison of Power Model with the SPICE simulations for CMOS 130nm

Circuit Element	SPICE (pW)	Power Model (without SCE) (pW)	Power Model (with SCE) (pW)	Error (without SCE)	Error (with SCE)
Inverter (2x)	372.7	901.2	411.8	141%	10.5%
4-Binary Tree	1156	1352	1212	16.9%	4.8%
Buffered Switch	883.2	1403	873.4	58.8%	1.1%

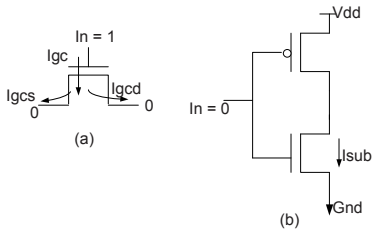

Figure 2: (a) Gate leakage in NMOS (b) Subthreshold leakage in Inverter

3.2 Leakage in FPGA Circuit Elements

This section describes various leakage current components that we have modeled in different circuit elements. The inverters were sized for equal rise and fall times, and for minimizing the delay and area product [4]. All the multiplexers were implemented with minimum sized inverters whereas, the SRAM cells are considered to have minimum sized transistors with high-V_{th} to mitigate subthreshold leakage, and the routing switches were optimized for area and delay product. We consider both the PMOS and NMOS in various circuit elements as the candidates for subthreshold leakage, but we consider only the NMOS transistors as candidates for gate leakage because the gate leakage in PMOS is considerably smaller than NMOS [10]. Furthermore, the back gate leakage of the NMOS transistors is ignored and we consider the gate current only from the gate to channel which then gets partitioned, and flows into the source and the drain of the transistor as shown in Fig. 2(a). We describe below the methodology we adopted for computing the leakage power for each of the circuit elements in the FPGA.

Inverter: We model the subthreshold leakage of the inverters in both the states, i.e, when the input is 0 and when the input is 1 and the gate leakage of the inverter when the input is 1. With the input at 0, only subthreshold leakage flows through the NMOS of the inverter and the gate leakage through PMOS is ignored as shown in Fig. 2(b). When the input to the inverter is 1, there is subthreshold leakage through the PMOS and gate leakage through the NMOS.

Multiplexer: In FPGAs, the multiplexers are implemented with NMOS pass transistor structures. The multiplexer is binary tree implemented using pass transistors. The leakage currents in the multiplexer is again strongly dependent on the state of its inputs. We analyze the multiplexer leakage with two cases as follows.

Case1: Fig. 3 shows the structure of the multiplexer and the subthreshold and gate leakages for the select signal (0,0) and the input vector (0010). Only one transistor (Q3) has subthreshold leakage, whereas three transistors have gate leakage currents (Q2,Q4,Q6). However, when the input vector changes to (0110), keeping the select signal same, there are three transistors which have subthreshold leakage (Q1,Q3,Q5) and two transistors have gate leakage(Q1,Q6). Therefore it is quite important to consider the state dependency of leakage currents in any circuit.

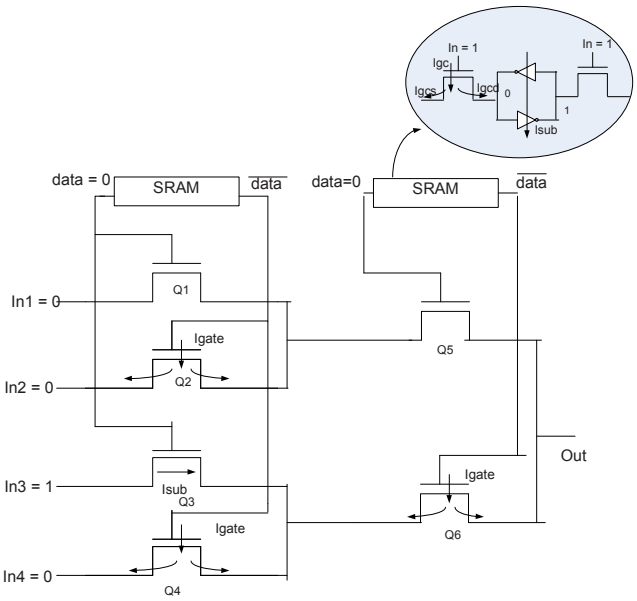

Figure 3: Multiplexer structure and the corresponding state dependent leakage for a particular select signal and input vector

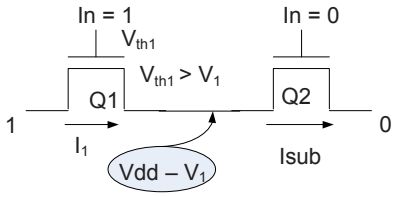

Figure 4: Leakage in multiplexers is affected by the voltage drop during signal propagation

Case2: Another phenomenon that needs to be accounted for in the pass transistor structures is that of the impact of V_{ds} on the threshold voltage of the transistor. Consider the case of two cascaded pass transistors as shown in Fig. 4. Here, transistor Q2 has subthreshold leakage. However, the drain terminal of Q2 is not at V_{dd}, but at a smaller value, which is $V_{dd} - V_1$, where V_1 is voltage which is smaller than the threshold voltage of Q1, (V_{th1}). This reduced drain voltage increases the threshold voltage of transistor Q2, which reduces the subthreshold leakage through Q2. It is interesting to note that $V_1 < V_{th1}$. This can be explained as follows. When Q1 tries to charge the drain node of Q2, Q1 has to be turned on, which implies that initially $V_1 > V_{th1}$ and Q1 is on and charges the node till $V_1 = V_{th1}$. After this, Q1 is turned off and subthreshold leakage current through it charges the drain node of Q2. At steady state Q1 needs to supply only the subthreshold leakage cur-

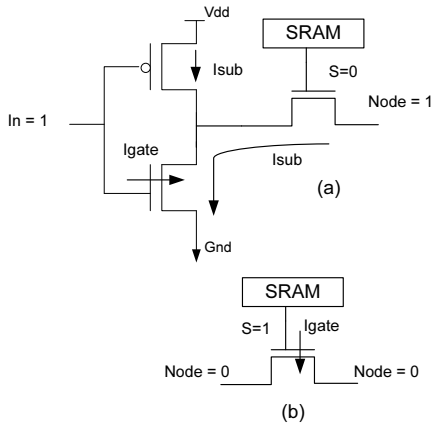

Figure 5: (a) Buffered routing switch. State dependent subthreshold and gate leakage currents. (b) Pass transistor routing switch. Only gate leakage is present when the switch is turned on.

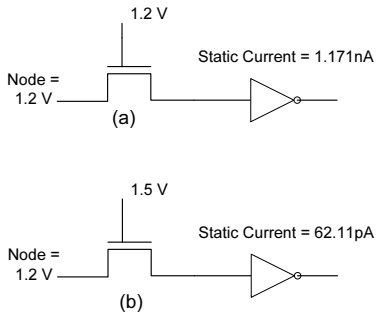

Figure 6: (a) Static current without gate boosting. (b) Reduced static current with gate boosting.

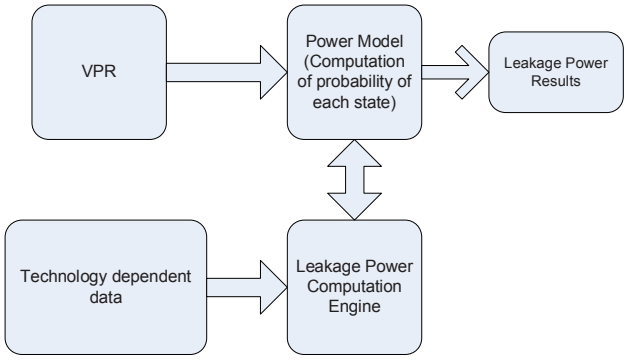

Figure 7: Overall architecture of the leakage power model

rent which is flowing through Q2. Under this condition, Q1 need not be turned on fully, i.e., it can operate in the subthreshold region and still provide sufficient current for the leakage current through Q2. Hence a steady state is reached when the voltage drop across Q1 is adequate to provide the necessary current. V_1 was assumed a constant value of $0.2V$, and $0.1V$ for CMOS 130nm and CMOS 90nm respectively. These values have been arrived at, using SPICE simulations and provide sufficiently accurate results. The leakage value reported in Table 1 for the 4 input binary tree takes this value of V_1.

SRAM Cells: The FPGA contains many SRAM cells which are used for configuring the FPGA. These SRAM cells are configured only once and it remains constant throughout the run time of the FPGA. We consider the standard six transistor SRAM cell. The SRAM cells are implemented with high-V_{th} transistors, because the SRAM cells are used only in the read mode, and is configured only once, and hence does not result in any performance penalty. This reduces the subthreshold leakage significantly and many commercial FPGAs have high-V_{th} SRAM cells. We model the leakage through two inverters connected back to back and gate leakage through one of the access pass transistors.

LUTs: The look-up tables (LUTs) consist of an array of SRAM cells and a multiplexer. The array of SRAM cells implement the truth table and the multiplexer selects the SRAM cell based on the input to the LUT. The leakage in this case would again be state dependent for the LUT as explained above for the multiplexers and inverters.

D Flip-flop: The D flip-flops are again made of latches and pass transistors so the leakage current for the flip-flops can also be modeled in terms of the basic inverter and pass transistors with the appropriate sizes of the transistors.

Routing Switches: There are two kinds of routing switches that are present in this FPGA architecture, namely, buffered routing switches and pass transistor based routing switches. Both switches have NMOS pass transistors. Fig. 5(a) shows the leakage currents through this switch when it is turned off with the input node at logic 1 and output node also at logic 1. In this case there is subthreshold leakage through the PMOS of the inverter and through the pass transistor of the switch. Fig. 5(b) shows the gate leakage current that flows through the pass transistor switch when the switch is turned on, and logic 0 is being passed through the switch.

The pass transistors in the routing switches have to drive buffers at the end of routing segments. When logic 1 is being driven through a NMOS pass transistor, it leads to a V_{th} drop in the voltage level of the signal. This leads to both the PMOS and NMOS of the driven buffer to get partially turned on leading to large static current. To address this problem commercial FPGAs employ gate boosting of the NMOS pass transistors to decrease the static current dissipated in the buffer driven by the NMOS pass transistor as depicted in Fig. 6. In this case the gates of the NMOS pass transistors are driven by a higher input voltage. Fig. 6 shows that the static current gets reduced considerably when gate boosting is employed.

4 Leakage Power Model

The overall architecture of the leakage power model is shown in Fig. 7. We use the widely used academic and research tool VPR [4], for placement and routing of the benchmark circuits. After the placement and routing of the given circuit, the power model computes the probability of states for each node of the circuit. For computing the probability of the nodes of the circuit, we have used the work done in [5]. The probability for each of the nodes is computed by propagating the static probability at the input, which are considered to be independent. The probability of any signal for a boolean function can be computed using the binary decision diagrams (BDD) [11]. Binary decision diagrams represent a logic

function graphically. A function $f(x_1, ..., x_n)$ can be written as

$$f = x_i.f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n) + \overline{x_i}.f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \quad (11)$$

using *Shannon's expansion*, where

$$f_{xi} = f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n) \quad (12)$$

$$f_{\overline{xi}} = f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \quad (13)$$

are the $cofactors$ of f. An input x_i is represented by a node in the BDD, and the edge coming out of the node represent the value of the input x_i. The value of the function can be determined by simply traversing the BDD from its root. The calculation of probability then becomes [11]

$$P(f) = P(x_i).P(f_{xi}) + P(\overline{x_i}).P(f_{\overline{xi}}) \quad (14)$$

Starting with $i = 1$, a depth first traversal of BDD would yield the the probability $P(f)$. After the probability of states for each of the node is computed, the power model looks at each of the circuit elements of the FPGA and computes the leakage for each of the states of that element using the Leakage Computation Engine (LCE). We have implemented the LCE, which computes the leakage for each of the circuit element of the FPGA based on given input vector (and the state of the SRAM cells, if they are present in the given circuit element). The LCE is basically a library of state dependent leakage calculation functions. This library has the basic leakage equation for subthreshold leakage and gate leakage and the computation of the associated parameters. It also has the models for computing the leakage for each of the FPGA circuit elements for a given input vector. For the used part of the logic block, we consider the actual probability of states depending on the input probability. For the unused part of the logic block, we consider that all the SRAM cell configuration bits are programmed to zero and compute the probability of states accordingly. In case of used pass transistor switches, they consume only gate leakage. The used buffered switches have subthreshold leakage power in the buffers and gate leakage in the pass transistor. For unused switches we consider that all the switches have different logic level at their two nodes. In this case, the buffers have both the gate and subthreshold leakage, whereas the pass transistors have only the gate leakage.

The leakage power model takes into account the state dependency of leakage power by considering the probability of states for each of the circuit element. Consider a circuit element which has n states and the probability of the states are $Prob_1$,$Prob_2$, ..., $Prob_n$, such that $\sum_{i=1}^{n} Prob_i = 1$. The leakage power for different states are given as $Pleak_1$, $Pleak_2$, ..., $Pleak_n$. The average leakage power can then be written as:

$$P_{avgleak} = \sum_{i=1}^{n} Prob_i.Pleak_i \quad (15)$$

5 Results and Discussion

For evaluating the leakage power consumption of different benchmarks, a fixed FPGA architecture for the benchmarks was taken. Smaller benchmarks had 20x20 logic blocks and a routing channel width of 100. Each logic block is made up of a cluster of 12 sub-blocks. For the larger benchmarks (bigkey, des, dsip), a square array of 35x35 logic blocks and a routing channel width of 100 was assumed.

Table 2: Subthreshold and gate leakage for different benchmarks

Benchmark	Subthreshold Leakage		Gate Leakage		Total Leakage	
	130nm (μW)	90nm (μW)	130nm (pW)	90nm (μW)	130nm (μW)	90nm μW)
alu4	138.34	596	498	21.7	138.34	617
apex2	152.37	667	502	21.6	152.37	689.46
apex4	131.6	567	509.6	22.1	131.6	589
bigkey	342.5	1455	1445	64.1	342.5	1519
des	347.7	1482	1447	64	347.7	1546
diffeq	140.5	603	486	21.1	140.5	624
dsip	332.3	1403	14351	63.7	332.3	1467
elliptic	187.2	844	517	21.8	187.2	866
ex1010	195.8	912	584	24.2	195.8	936
ex5p	126.6	541	505	22	126.6	563
frisc	185.3	838	525	22.1	185.3	860
misex3	136.5	588	498	21.6	136.5	609
s298	156.7	679	479	20.8	156.7	700
spla	177.5	659	549	21.5	177.5	681
tseng	128.8	547	490	21.4	128.8	569

Table 2 shows the leakage power consumption for different benchmarks. It can be seen that the dominant leakage for both the technologies is the subthreshold leakage. The gate leakage is orders of magnitude smaller than the subthreshold leakage for CMOS 130nm. The gate leakage is less than $1/20^{th}$ of the subthreshold leakage for the CMOS 90nm. The subthreshold leakage for the CMOS 90nm FPGA is almost 4 times greater than the subthreshold leakage for the CMOS 130nm FPGA. Further, it is evident that the gate leakage increases exponentially with technology scaling. For the CMOS 90nm FPGA, the gate leakage is orders of magnitude greater than the gate leakage for CMOS 130nm FPGA. This result is consistent with the fact that the contribution of the gate leakage to total leakage increases with technology scaling. The state dependency of the leakage is evident from the fact that the leakage power for different benchmarks are different, even though the same size of FPGA is considered for benchmarks. It should be noted that the total SRAM leakage remains constant for all the benchmarks as it is dependent only on the total number of SRAM cells and since we used a fixed size FPGA for all the benchmarks, the SRAM leakage remains constant for all the implementations. However, the leakage, especially in the logic part is strongly state dependent. For example, the logic leakage in case of the benchmark spla is 72μW, whereas in case of ex5p is 35.72μW, almost half of the logic leakage of spla (CMOS 130nm). For the CMOS 90nm the logic leakage for spla and ex5p are 368μW, and 208μW, respectively, again showing a lot of dependency on state.

Fig. 8 shows the distribution of average leakage power in different parts of the FPGA for the CMOS 130nm and CMOS 90nm. The SRAM leakage is very small as compared to the logic and the routing leakage because the SRAM cells are implemented with high-V_{th} transistors. It can be seen that the dominant leakage is the routing leakage for both CMOS 130nm and CMOS 90nm. However, the contribution of logic leakage to the total leakage increases from 33.6% to 41.5% when the technology is scaled from CMOS 130nm to CMOS 90nm. This is because the V_{th} of the PMOS

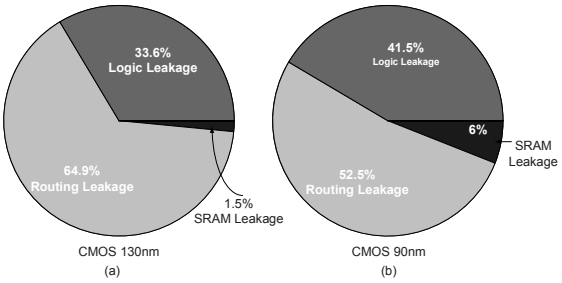

Figure 8: Average Leakage distribution for different parts the FPGA for CMOS 130nm and 90nm

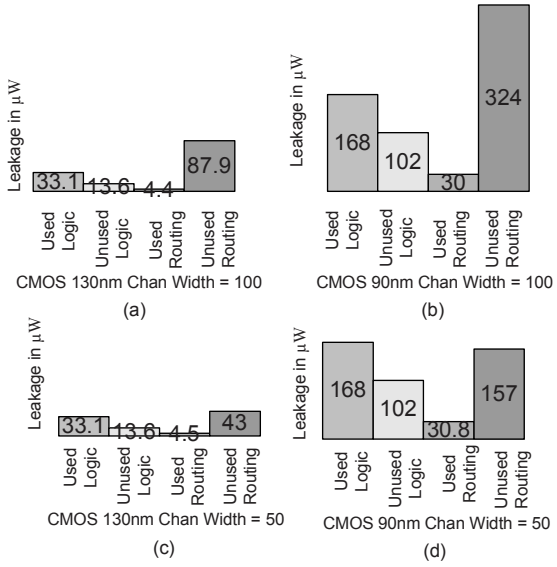

Figure 9: (a),(b)Used and unused leakage for different components of FPGA for the benchmark alu4 for the FPGA architecture with routing channel width of 100 (c),(d) With routing channel width of 50

is CMOS 90nm suffered from narrow width effect, which consequently reduced it, leading to increased contribution of leakage from the PMOS. Since most of the PMOS transistors are present in the logic part, the contribution of the logic leakage to the total leakage increased. It is evident from Fig. 9 that routing leakage is the dominant leakage power for the given FPGA architecture. Majority of the routing leakage power comes from the unused part of the routing resources. Further, for most FPGA designs the logic utilization is quite high, whereas the utilization of routing resources is quite low. In the initial case with the routing channel width of 100, the routing leakage was considerably larger than the logic leakage. Almost all the routing leakage comes from the unused routing resources, whereas the major part of the logic leakage is from the used logic part. However, when the routing channel width is reduced to 50 (alu4 can be placed and routed with a channel width of 50), the leakage from the routing resources reduces to almost half, as expected. This results in significant reduction of total leakage. The contribution of the logic leakage to total leakage becomes slightly more than the routing leakage for the CMOS 130nm. For CMOS 90nm the logic leakage clearly starts dominating the routing leakage, when the routing resources is reduced to half. However, the total logic leakage remains constant, as expected. This shows that FPGA CAD tools should try to increase the utilization of the routing resources, so that FPGAs can be implemented with lesser routing resources to reduce leakage.

6 Conclusions

In this paper we presented an analytical state dependent leakage power model for FPGAs. The leakage power model was developed based on physical and empirical equations for the devices, making it technology independent and can be used for rapidly analyzing different FPGA architectures across different technology nodes. We presented a leakage analysis for FPGAs in CMOS 130nm and CMOS 90nm. The results indicate that the leakage power in CMOS 90nm FPGA increases significantly over the CMOS 130nm FPGA, which is as expected. For our future work we intend to use the power model to develop techniques for total leakage reduction in FPGAs.

7 Acknowledgements

We would like to thank Widad Machmouchi for helping with parts of code for the leakage power model. This work was supported by NSERC.

References

[1] L. Wei et al. "Design optimization of dual-threshold circuits for low-voltage low-power applications," *IEEE Trans. VLSI, Vol. 7, pp. 16-24, March 1999.*

[2] T. Tuan et al., "Leakage Power Analysis of a 90nm FPGA," *IEEE Custom Integrated Circuits Conf., pp. 57-60, 2003.*

[3] F. Li et al., "Architecture evaluation for power efficient FPGAs," *FPGA'03, pp. 175-184, 2003.*

[4] V. Betz et al., *Architecture and CAD for Deep-Submicron FPGAs*, Kluwer Academic Publishers, MA 1999, ISBN: 0792384601

[5] K. Poon et al.,"A flexible power model for FPGAs," *International Conf. on Field Programmable Logic and Applications, 2002.*

[6] F. Li et al., "Low-power FPGA using predefined dual-Vdd/dual-Vt fabrics," *FPGA'04, pp. 42-50, 2004.*

[7] K. Roy et al.,"Leakage current mechanisms and leakage reduction techniques in deep-submicrometer circuits" *Proc. of the IEEE, Vol. 91, Feb 2003*

[8] BSIM4 Models, University of California, Berkeley. Available online: *http://www-device.eecs.berkeley.edu/ bsim3/bsim4.html*

[9] S. Sirichotiyakul et al., "Duet: an accurate leakage estimation and optimization tool for dual-Vt circuits" *IEEE Trans. on VLSI, Vol. 10, pp. 79-90, April 2002*

[10] D. Lee et al., "Gate oxide leakage current analysis and reduction for VLSI circuits" *IEEE Trans. on VLSI, Vol. 12, pp. 155-166, Feb. 2004*

[11] K. Roy and S. C. Prasad, *Low Power CMOS VLSI Circuit Design*, Wiley-Interscience, John Wiley & Sons, 2000, ISBN: 0-471-11488-X