

# An Adaptive FPGA Architecture with Process Variation Compensation and Reduced Leakage

Georges Nabaa<sup>‡</sup>, Navid Azizi<sup>†</sup>, Farid N. Najm<sup>†</sup>

<sup>†</sup>Actel, Mountain View, California

<sup>‡</sup>Department of ECE, University of Toronto, Toronto, Ontario, Canada

georges.nabaa@actel.com , {nazizi,najm}@eecg.utoronto.ca

## ABSTRACT

Process induced threshold voltage variations bring about fluctuations in circuit delay, that affect the FPGA timing yield. We propose an adaptive FPGA architecture that compensates for these fluctuations. The architecture includes an additional characterizer circuit that classifies logic and routing blocks on each die according to their performance. Based on this classification, the architecture adaptively body-biases these resources by either speeding up the slow blocks or by slowing down the leaky ones. This procedure mitigates the effect of the variations and provides a better yield. We further diminish leakage by slowing down areas of the FPGA that have a positive slack. Overall, this architecture minimizes the timing variance of within-die and die-to-die  $V_{th}$  variations by up to 3.45X and reduces leakage power in the non-critical areas of the FPGA by 3X with no effect on frequency.

## Categories and Subject Descriptors

B.3.1 [Integrated Circuits]: Types and Design Styles

## General Terms

Design

## Keywords

FPGA, Process Variations, Leakage, Body-biasing

## 1. INTRODUCTION

CMOS scaling has been driven by the desire for higher transistor densities and faster devices. Historically, innovations for improving performance relied on exploiting ever larger numbers of transistors operating at higher frequencies. To keep the resulting switching power dissipation at bay, successive technology generations have relied on reducing the supply voltage ( $V_{dd}$ ) and the minimum transistor

feature size. But while supply voltage reduction guarantees an acceptable dynamic power dissipation, it comes at the expense of overall performance. Hence, a corresponding decrease in the transistor threshold voltage ( $V_{th}$ ) is required to restore the current drive of the device. However, this reduction in  $V_{th}$  has not been accompanied by a similar drop in threshold voltage *variations* ( $\Delta V_{th}$ ) [1]. Furthermore, the reduction in the minimum feature size has caused significant variation in other transistor characteristics such as transistor length. This further aggravates  $\Delta V_{th}$  through short channel effects fluctuations [1]. The increased process variations may have a significant effect on circuit performance, power and yield [2]; process variations can cause up to 30% disparities in chip frequency and a 20x spread in subthreshold current [3]. While compensating for process variations has been an active research topic for the microprocessor and Application Specific Integrated Circuit (ASIC) design communities [3, 4], reducing the process variations in Field-Programmable Gate Arrays (FPGAs) has not been addressed. Furthermore, compensating for Within-Die (WID) process variations in ASICs increases the area considerably. Unlike ASICs, FPGAs offer regular structures that are well suited to compensation with an incremental algorithm.

We propose a new adaptive architecture that compensates for the process variations on FPGAs by including an additional circuit block per FPGA chip that incrementally characterizes each FPGA logic and routing block based on their performance. The classification is then stored in additional SRAM configuration bits in each tile which is used by a bidirectional adaptive body-bias circuit to compensate for process variations. Circuit blocks that have a higher than nominal performance, through a lower threshold voltage or shorter length, will be slowed down and made less leaky; conversely circuit blocks that are slower than nominal will be made faster. Furthermore our additions to the FPGA can be used to reduce the leakage of the FPGA by using the body-bias to selectively slow down logic and routing blocks that are not critical. To our knowledge, this is the first work that addresses process variations in the context of FPGAs. The resulting architecture offers two substantial improvements over conventional FPGA architectures:

1. Reduction of the effect of process variations. This will in turn reduce the corresponding delay variations on the FPGA. The reduction in delay variations allows for lower guard-banding and thus the ability to produce more aggressive designs.
2. A secondary effect of being able to reduce overall leak-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

DAC 2006, July 24–28, 2006, San Francisco, California, USA.

Copyright 2006 ACM 1-59593-381-6/06/0007 ...\$5.00.

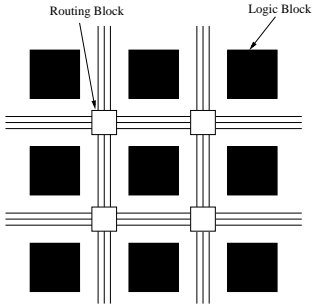


Figure 1: Standard FPGA Architecture

age by slowing down non-critical portions of the FPGA.

It will be shown that the variance of the process variations can be reduced by 3.45X and that leakage power can be reduced by 3X with no effect on frequency.

The remainder of this paper proceeds as follows. Section 2 provides background on FPGAs, commonly used techniques for leakage power minimization in FPGAs and process variation minimization in ASICs. Section 2 also underscores the shortcomings of these techniques, which provide the motivation for our approach. Section 3 discusses the new adaptive architecture, and the corresponding algorithm responsible for setting the additional configuration bits and a leakage optimization scheme based on our novel architecture. Sections 4 and 5 provide an overview of our methodology and the results respectively. Finally, 6 concludes the paper.

## 2. BACKGROUND

### 2.1 FPGAs

FPGAs are composed of many configurable logic blocks connected together through a configurable interconnect [5], as shown in Fig. 1 which depicts a standard island-style FPGA architecture. Logic blocks are clusters of logic elements, which are themselves composed of a single Look-up Table (LUT) and a register. The distributed interconnect is made up of routing switches which are buffered multiplexers that create a many to one input to output mapping. Both the LUT and the routing switch are programmable through configuration SRAM bits that are included with each block. In this paper a *block* refers to both logic and routing blocks unless otherwise specified.

### 2.2 Leakage Reduction in FPGAs

Process variations and leakage power are two challenges that are faced in sub-90nm technologies. Since subthreshold leakage has an exponential relationship to the threshold voltage,  $V_{th}$ , process variations can increase the leakage substantially. Previous work on leakage minimization in FPGAs ignored process variations. Such techniques include:

1. Dual  $V_{th}$  and Dual  $V_{dd}$  [6]
2. Programmable Dual  $V_{dd}$  [7]
3. Programmable Routing Switch [8]

[6] exploits the fact that the configuration cells can be slowed down and thus assigned a high  $V_{th}$  since they are used only once upon loading (no run-time delay penalty). Their approach also segregates predefined sections of the fabrics into

high and low  $V_{dd}$  sections where critical and non-critical logic can be programmed into. However, [6] highlights the limitations of their method by stating that a programmable  $V_{dd}$  solution would allow for higher flexibility, where a timing analysis tool adjusts the supply voltage for each block in the FPGA according to slack availability. Therefore a block on the critical path is allocated a high  $V_{dd}$  whereas blocks with a large slack are allocated a low  $V_{dd}$ .

[7] introduces such a heuristic and is able to achieve significant power gains (61%). In much the same way [8] develops a programmable routing switch which is placed in a low-power mode when there is available slack on the path. All these techniques, however, ignore process variations by assuming that the available slack estimated by the timing analysis tool for a given block is equal to the post-fabrication timing slack for that block. This is, however, not the case since the timing slack in the design phase can be significantly different from the silicon measurements, due to within-die and die-to-die process variations. Therefore, applying [7]’s leakage minimization techniques on a block could lead to functional errors due to failure in meeting timing constraints. To cope with this intrinsic variability, designers have considered worst-case process variations thus limiting the performance and leakage reduction possible. Our approach starts by addressing process variations before applying the leakage optimization schemes thus allowing for more aggressive leakage reduction.

### 2.3 Process Variation Compensation

A number of recent studies have considered the use of adaptive body biasing as a way of compensating for die-to-die and within-die parameter variations [3, 4]. In one method the critical path is replicated and used to characterize the rest of the die [4]. This approach is aimed at ASICs since it requires prior knowledge of the design’s critical path which is not available when fabricating an FPGA. Moreover, the technique requires multiple duplicates of the critical path circuit to compensate for WID variations [4]; to compensate for WID variations, the die is divided into small blocks and a circuit which is representative for that block is replicated in each block. The technique works under the assumption that transistors within the block will be effected by similar process variations. Since the technique needs to replicate many different critical paths, using a large number of blocks would lead to an excessive increase in area. Furthermore prior knowledge of the critical path needs to be known which is not the case in an FPGA where the critical path differs from design to design. Also, unlike ASICs, FPGAs offer regular structures that are well suited compensation with an incremental algorithm.

## 3. ADAPTIVE ARCHITECTURE

The new adaptive architecture is composed of three additional circuits on the FPGA; one characterizer per FPGA, extra configuration SRAM bits per FPGA tile, and bi-directional body-bias generators per FPGA tile. The location of the characterizer in the FPGA fabric is shown in Fig. 2. The characterizer is responsible for measuring the process variations that have occurred in each FPGA blocks. When an FPGA is powered ON the characterizer classifies each logic and routing block into the following many different groups such as:

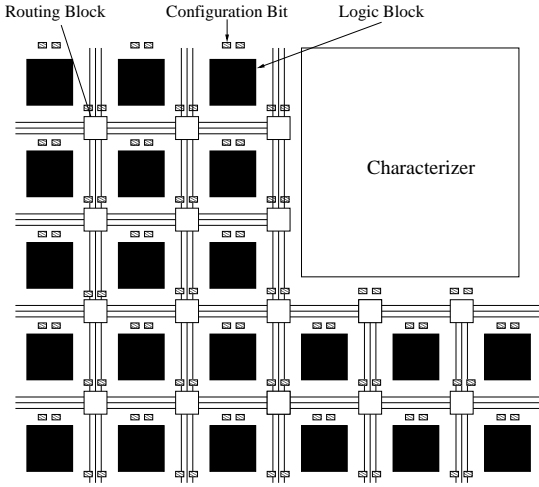


Figure 2: Proposed FPGA Architecture (Island-style)

1. High  $V_{th}$ , low leakage, low speed
2. Nominal  $V_{th}$ , nominal leakage, nominal speed
3. Low  $V_{th}$ , leaky, faster speed

This classification is then stored in SRAM configuration bits in each tile which then select a body-bias value which is routed to each tile to adjust the threshold voltage,  $V_{th}$ , of the block and to compensate for the underlying process variations.

The number of groups that the categorizer can classify each logic and routing block into, and consequently the number of additional configuration SRAM bits that are needed to allow for the different levels of compensation, provides a tradeoff between the area overhead and the amount of compensation that is possible. Additional compensation levels allow for a smaller resulting variation at the expense of a more complex biasing circuit and larger area overhead. In this paper we consider 3- and 7-level classification which correspond to 2 and 3 additional configuration bits per FPGA block respectively.

Since additional SRAM bits and the bidirectional adaptive body-biasing circuits [4] are well known designs the rest of this section will provide the details of the characterizer block.

The block diagram of the characterizer is shown in Fig. 3. The characterizer is composed of a phase detector, a sample and hold circuit, a comparator and high level controller. The characterizer works as follows: first a clock signal (CLK) is fed into the phase detector as well as being routed to the FPGA block that is being characterized. The output of that block,  $CLK'$  is fed into the second input of the phase detector. The phase detector compares the two clock signals and produces an output whose pulse width represents the delay associated with going through the characterized block. The pulse width can be described as:

$$pulseWidth = d + \delta_d \quad (1)$$

where  $d$  is the nominal delay involved in traversing a block and  $\delta_d$  is the variation in delay of the block due to process variations. For example, if the  $V_{th}$  for the block is higher than nominal or the length is longer than nominal, then  $\delta_d$

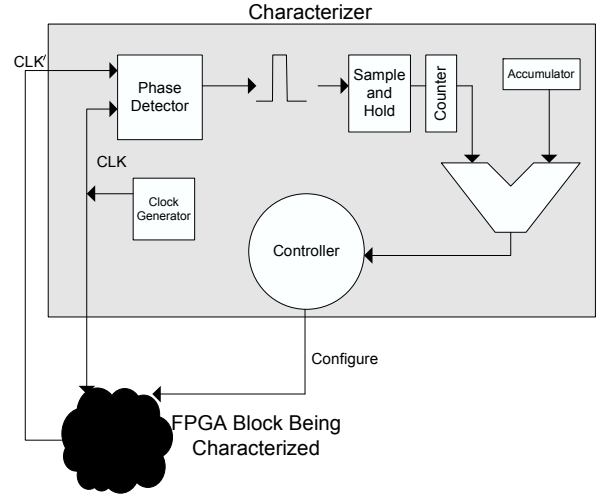


Figure 3: Characterizer High-level Block Description

will be positive representing a slow block. Conversely if the  $V_{th}$  for the block is lower than nominal or has a narrower length, then  $\delta_d$  will be negative representing a fast but leaky block.

The analog pulse width is then converted into a digital value by sampling the analog pulse using a high frequency clock  $CLK_2$ . The sampled value is stored in a counter. Now the value  $d + \delta_d$  is represented by a digital value and can be compared to the expected delay  $d$ . The output of this comparison will set the configuration bits in each FPGA block. In turn these configuration determine the amount of body-bias to be applied.

The algorithm described above is only valid when used on logic or routing blocks that are adjacent to the characterizer. However, it can easily be extended to all the blocks in the FPGA. The key idea is that since the characterizer goes through the blocks sequentially, it keeps track of the delays incurred when traversing previously calibrated blocks. Hence, when a new resource is reached, the intermediate interconnect delay required to reach that block is already known and can be subtracted out. Hence, the delay of a distant block can be isolated and the methodology that was used for blocks contiguous to the characterizer can now be extended to all the blocks in the FPGA. For example, assume that the clock loop-back signal has to go through  $n - 1$  routing and logic blocks before reaching the one that is being characterized. Therefore the pulse width at the output of the phase detector now represents:

$$pulseWidth = \sum_{k=1}^n d + \delta_d \quad (2)$$

The variation  $\delta_d$  results only from the block being characterized, because the sequential property of our algorithm ensures that the delay of the  $n - 1$  blocks that the clock signal traverses, are already known. Since the variability in the previous blocks is not completely removed, the controller has to also store the intermediate post-calibration block delays in order to perform an accurate characterization of the distant blocks. With this knowledge the characterizer sam-

ples the output of the phase detector into a digital counter as before and subtracts the known delay thus obtaining  $\delta_d$ . As before, depending on the sign and value of  $\delta_d$ , the configuration bits in the FPGA block are set and the resulting proper compensation will be applied.

### 3.1 Leakage Optimization

An additional benefit of this architecture is the ability to optimize for leakage and/or speed. Previous work has shown that FPGA designs have a considerable amount of timing slack that can be used to reduce the leakage. It has been demonstrated that on average 75 % of all routing resources can be slowed down by 50% [8].

This observation allows our architecture to have an additional benefit of being able to reduce the leakage. Sections of designs that have a timing slack can be slowed down by overwriting the configuration bits of their routing and logic blocks thus raising their threshold voltage and reducing their leakage. This can be safely done since variability has already been compensated for during the calibration process.

## 4. METHODOLOGY

All simulation results reported in this paper are based on an industrial 130nm technology and are produced using HSPICE. All simulations presented were performed at 110°C where leakage and delay are more critical than at low temperatures.

### 4.1 Variation Modeling

Variations normally have three components: a die-to-die component, a within-die systematic component, and a within-die random component. The term “systematic” refers to the parts of the variations which have some correlation across the die, while the “random” component refers to the parts of the variations that are totally independent. In this paper we focus on the WID random component of the threshold voltage with no loss of generality to different sources of variation since our architecture compensates by measuring the effect of variations, and not the underlying variation itself. The variations are modelled as a standard normal distribution.

The process variations for each transistor are obtained as follows. First the the  $3\sigma$  values of the normal distribution for minimum sized transistors were obtained by the worst case values from the technology file, then the variance for each transistor is obtained by scaling the  $3\sigma$  value according to the area of transistor [9, 10] where the magnitude of  $V_{th}$  variations is inversely proportional to the square root of transistor area. The  $V_{th}$  variation obtained is then assigned to the HSPICE DELVTO parameter for each transistor. DELVTO is a user-specified HSPICE parameter that shifts the value of the threshold voltage during simulations.

### 4.2 Architecture Modeling

In order to verify that our architecture reduces the effect of variation on performance, we prototyped our new architecture. First the process variations are generated from the normal distribution and are simulated in SPICE. The output of this SPICE simulation is input into a module that models the behaviour of the characterizer circuit. The characterizer uses the data from SPICE to compute the configuration SRAM bit settings which are fed back into the SPICE simulation for a second iteration. The performance and leakage results of the second SPICE simulation represent the results

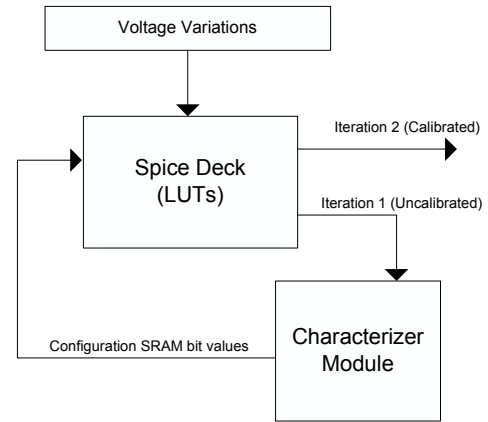


Figure 4: Simulation Architecture

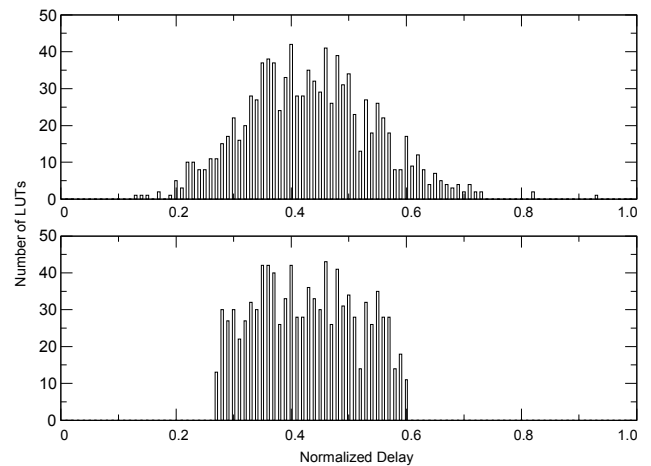


Figure 5: Variation in Delay for the Standard Architecture (above) and the Adaptive Architecture (below) for 3 levels of classification

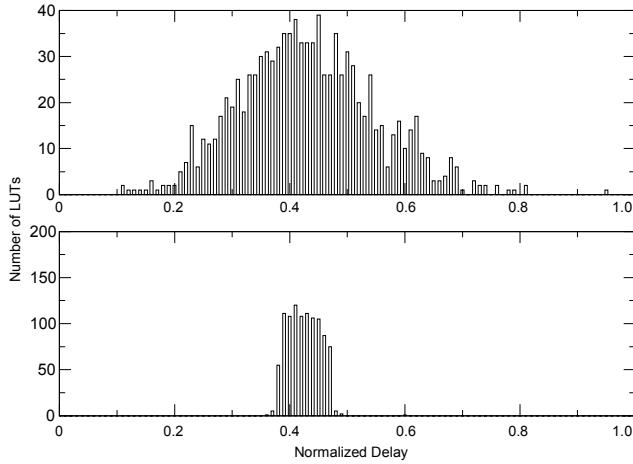
for our proposed architecture. Fig. 4 depicts our simulation methodology.

## 5. RESULTS

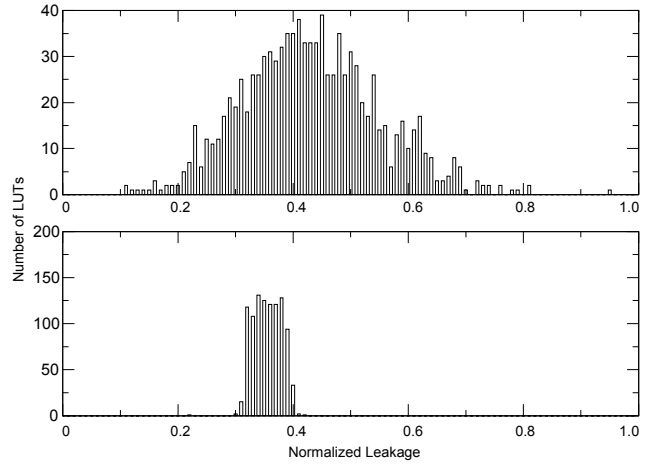
The following figures compare the leakage and timing behaviour of LUTs across an FPGA which have threshold voltage variations applied to them in a standard architecture and the adaptive architecture. All the LUTs are configured as 2-input NANDs. The results are shown for both 3- and 7-levels of classification.

Fig. 5 show the effect of the proposed architecture on the variation in delay when using three levels of classification. While the mean has not changed, the standard deviation in delay shows a decrease of 30.3%. Fig. 6 shows the variation in delay when using seven levels of classification; in this case the standard deviation in delay shows a reduction of 3.3X.

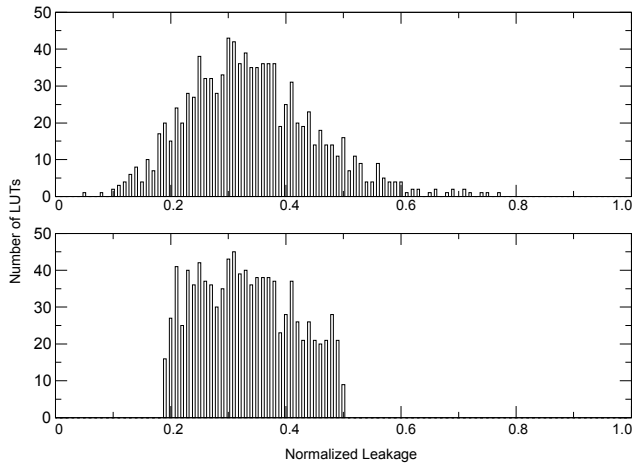
In terms of the effect of the new architecture on the variation in leakage, Fig. 7 shows the range of leakages when using three levels of classification. Again while the mean leakage has not changed, the standard deviation of the leak-



**Figure 6: Variation in Delay for the Standard Architecture (above) and the Adaptive Architecture (below) for 7 levels of classification**



**Figure 8: Variation in Leakage for the Standard Architecture (above) and the Adaptive Architecture (below) for 7 levels of classification**



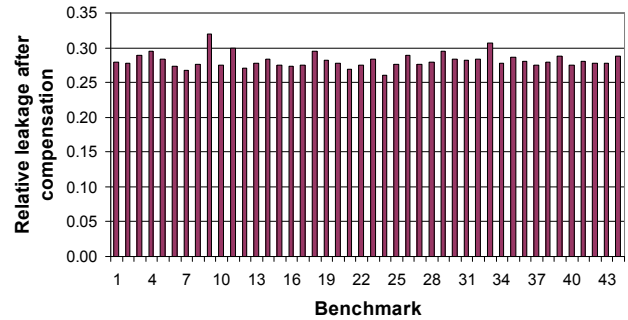
**Figure 7: Variation in Leakage for the Standard Architecture (above) and the Adaptive Architecture (below) for 3 levels of classification**

age has been reduced by 78%. The reason why there is a larger reduction in the standard deviation of the leakage compared to the reduction in the standard deviation of the delay is that leakage has an exponential relationship to the transistor threshold voltage.

When using a seven-level classifier an 18X reduction in the standard deviation of the leakage is observed.

### 5.1 Leakage Reduction

We have also analyzed how much leakage can be reduced by using the inherent body-biasing in the new architecture by slowing down areas of the FPGA that have slack. We used SmartTime, Actel’s Static Timing Analysis tool to extract the slacks for 44 industrial designs placed and routed on the Pro-Asic3 architecture. Based on the slacks, we increase the threshold voltage of these cells and achieve leakage reduction with no additional performance penalty.



**Figure 9: Reduction in leakage by applying Compensation Scheme**

Therefore, all 44 designs still meet their timing constraint targets following the threshold voltage slow down.

Fig. 9 shows the leakage values for the suite of designs normalized against a standard architecture. These results show a nearly uniform distribution in which leakage measurements using our architecture are 30% those of the standard architecture. The reduction in the leakage is in line with those found by [8].

### 5.2 Area

The new adaptive FPGA architecture has an increased area and cost. There is an increased cost and area overhead due to the need for a triple-well process, but since routing switches and LUTs are mostly comprised of NMOS pass-transistors the area overhead for triple-well process will be less than other body-biasing schemes. The extra transistor overhead in each FPGA tile includes an additional two or three SRAM bits and an extra multiplexer to choose the select the appropriate body-biasing value which is around 1.6% of the transistors in the tile. Furthermore, there is the area overhead of the characterizer which we estimate to be less than the area of nine FPGA tiles. This area penalty is reasonable considering the thousands of tiles that make up

modern FPGAs.

## 6. CONCLUSION

In this paper, we proposed a novel adaptive FPGA architecture that reduces the effect of process variations on the delay and leakage of FPGA circuit blocks. Our adaptive FPGA architecture adds one characterizer circuit per FPGA along with body-bias generators throughout the FPGA fabric and uses an iterative algorithm on start-up to compensate for within-die and die-to-die threshold voltage variations. An area/optimization trade-off exists; if three levels of classification are used the delay variability is reduced by 30% and the leakage variability is reduced by 78%. On the other hand if a seven level classification is used, the delay variability is reduced by 3.3X and the leakage variability is reduced by 18X. Moreover the new adaptive architecture allows for FPGA blocks which are not critical to be slowed down, hence reducing the leakage while meeting the timing design constraints. Simulations show that for an Actel ProAsic3 architecture, on a benchmark set of 44 industrial design, a 3X decrease in leakage is achieved.

## Acknowledgments

The authors would like to acknowledge the technical guidance and support of Nizar Abdallah and Amal Zerrouki at Actel Corporation

## 7. REFERENCES

- [1] S. Nassif. Delay variability: Sources, impacts, trends. *ISSCC*, 2000.
- [2] S. Borkar, T. Karnik, S. Narendra, T. Tschanz, A. Keshavarzi, and Vivek De. Parameter variations and impact on circuits and microarchitecture. *Design Automation Conference*, 2003.
- [3] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Kesha varzi, and V. De. Parameter variations and impact on circuits and microarchitecture. In *ACM/IEEE 40th Design Automation Conference (DAC-03)*, pages 338–342, Anaheim, CA, June 2-6 2003.
- [4] J.W. Tschanz, J.Y. Kao, and S.G. Narendra et al. Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. In *IEEE Journal of Solid State Circuits*, pages 1396–1402, November 2002.
- [5] Vaughn Betz, Jonathan Rose, and Alexander Marquardt. *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers, 1999.
- [6] F.Li., Y.Lin, and J.Cong. Low power fpga using pre-defined dual-vdd/dual-vt fabrics. In *ACM International Symposium on Field Programmable Gate Arrays*, pages 42–50, Monterey, CA, USA, February 22-24 2004.
- [7] A.Gayasen, K.Lee, N.Vijaykrishnan, M.Kandemir, M.Irwin, and T.Tuan. A dual vdd low power fpga architecture. In *Proceedings of the International Symposium on Field Programmable Gate Arrays*, pages 51–58, August 2004.
- [8] J.H. Anderson and F.N. Najm. Low-power programmable routing circuitry for fpgas. In *International Conference on Computer Aided Design (ICCAD 04)*, pages 602–608, San Jose, CA, November 7-11 2004.
- [9] U. Grunebaum, J. Oehm, and K. Schumacher. Mismatch modeling and simulation- a comprehensive approach. In *Analog integrated circuits and signal processing*, pages 165–171, December 2001.
- [10] S.J. Lovett, L. Wall, and M. Welton et al. Sensitivity of MOS transistor mismatch to device dimensions and suggestions on how to improve matching performance. In *Improving the efficiency of IC manufacturing, IEEE Colloquium on*, pages 11/1–11/5, London, UK, April 12 1995.