LNCS 7606

José L. Ayala
Delong Shang
Alex Yakovlev (Eds.)

Integrated Circuit and System Design

Power and Timing Modeling, Optimization and Simulation

22nd International Workshop, PATMOS 2012
Newcastle upon Tyne, UK, September 2012
Revised Selected Papers

Springer

Lecture Notes in Computer Science 7606

José L. Ayala   Delong Shang
Alex Yakovlev (Eds.)

Integrated Circuit and System Design

Power and Timing Modeling, Optimization and Simulation

22nd International Workshop, PATMOS 2012
Newcastle upon Tyne, UK, September 4-6, 2012
Revised Selected Papers

Springer

Volume Editors

José L. Ayala
Complutense University of Madrid
Facultad de Informática, 28040 Madrid, Spain
E-mail: jayala@fdi.ucm.es

Delong Shang
Newcastle University
School of Electronic Engineering
Newcastle upon Tyne, NE1 7RU, UK
E-mail: delong.shang@ncl.ac.uk

Alex Yakovlev
Newcastle University
School of Electronic Engineering
Newcastle upon Tyne, NE1 7RU, UK
E-mail: alex.yakovlev@ncl.ac.uk

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-36156-2 e-ISBN 978-3-642-36157-9
DOI 10.1007/978-3-642-36157-9
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012955597

CR Subject Classification (1998): C.4, I.6.3-7, C.2, B.5.2, B.6.3, B.7.2, D.3.4

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

PATMOS 2012 was the 22nd in a series of international workshops on Power and Timing Modeling, Optimization and Simulation. The PATMOS meeting has evolved, during the years, into a leading scientific event where industry and academia meet to discuss power and timing aspects in modern integrated circuit and system design. Both universities and companies are invited to participate.

The objective of this workshop is to provide a forum in which to discuss and investigate emerging challenges in methodologies and tools for the design of upcoming generations of integrated circuits and systems, including reconfigurable hardware such as FPGAs. The technical program focused on timing, performance, and power consumption as well as architectural aspects with particular emphasis on modeling, design, characterization, analysis, and optimization.

September 2012

Alex Yakovlev
Delong Shang

Organization

PATMOS 2012 was organized by Newcastle University, UK.

Organizing Committee

General Chairs

Alex Yakovlev	Newcastle University, UK
Delong Shang	Newcastle University, UK

Program Chair

José L. Ayala	UCM, Spain

Publicity Chair

Ian Clark	Newcastle University, UK

Publication Chair

Fei Xia	Newcastle University, UK

Local Committee

Joan Atkinson
Maciej Koutny
Claire Smith
Danil Sokolov

Steering Committee

Antonio J. Acosta
Nadien Azemard
Joan Figueras
Reiner Hartenstein
Jorge Juan-Chico
Enrico Macii
Philipe Maurien
Jose Monteiro
Wolfgang Nebel
Vassilis Paliouras
Christian Piguet
Dimitrios Soudris
Diederik Verkest
Roberto Zafalon

Sponsoring Institutions

Newcastle University
University of Leicester
EPSRC
Formal Methods Europe

Table of Contents

Sleep-Transistor Based Power-Gating Tradeoff Analyses*

Sven Rosinger[1] and Wolfgang Nebel[2]

[1] OFFIS Research Institute
[2] University of Oldenburg,
D - 26121 Oldenburg, Germany
rosinger@offis.de, nebel@informatik.uni-oldenburg.de

Abstract. Power-gating is a promising leakage-reduction technique in recent sub-100*nm* semiconductor technologies but its efficiency and usability depends on several parameters. Beside the technology node size and related parameters (e.g. process corner) it also depends on the switch implementation scheme (e.g. header vs. footer device, single vs. double cutoff), the sleep transistor sizing, and on dynamic parameters such as the supply voltage. In this work, typical sleep-transistor based power-gating schemes are applied to RT-level components and leakage reductions, break-even-, and wake-up-times are traded off for relevant parameters and possibilities as well as limitations of these schemes are evaluated. It is shown that the break-even time varies up to a factor of 4 and the wake-up time up to a factor of 6 solely due to the power gating scheme selection while the leakage reduction is above 95% in all cases.

1 Introduction

The international technology roadmap for semiconductors (ITRS) working group identified *dynamic power-management* to be one of the most promising challenges to counteract the power issue of today's high performance integrated circuits [7].

Within this class of techniques, sleep-transistor based power-gating has evolved to be the most effective leakage-reduction technique. By applying a header or footer device in series to a circuit block these blocks can be powered down for times of disuse and the leakage currents are reduced effectively.

Figure 1 schematically illustrates such a power-down sequence of a circuit block with a p-type header device. The sequence is divided into the phases $active_1$, *power down*, *sleep*, *wake up*, and a secondary active phase $active_2$.

Thereby a sleep phase needs to be long enough in order to amortize the state transition energy E_{SW} that occurs due to capacitance charging during wakeup as it is defined in Equation 1.

* This work was supported by the European Commission within the Seventh Framework Programme through the European Integrated Project COMPLEX under Grand Agreement 247999.

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 1–10, 2013.

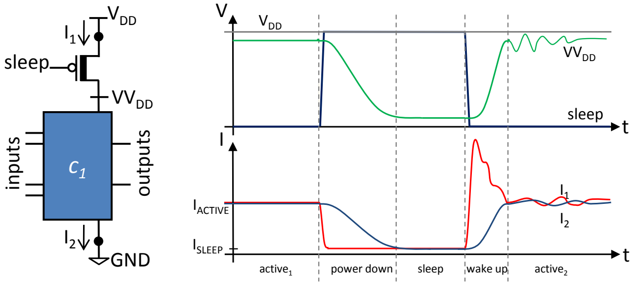


Fig. 1. p-type power-gating functionality

$$E_{SW} = \int_{t_{wakeup}} V_{DD} \cdot I_1(t) \tag{1}$$

With known leakage currents in active and sleep state (I_{ACTIVE} and I_{SLEEP}) a break-even time t_{be} can be computed with Equation 2 that defines the minimum time beeing powered down to compensate state transition costs. Thereby, some further costs need to be considered because the header or footer device, necessary interfacing circuits as voltage anchors, or buffers have inherent leakage currents and consume an inherent amount of energy during state transition that are summarized as $I_{OVERHEAD}$ and $E_{OVERHEAD}$.

$$t_{be} = \frac{E_{SW} + E_{OVERHEAD}}{(I_{ACTIVE} - (I_{SLEEP} + I_{OVERHEAD})) \cdot V_{DD}} \tag{2}$$

The most important questions in applying power-gating include the technical implementation of the gating switch, its size, and its temporal and spatial granularity of application. In this work, power-gating is applied to functional units at RT-level such as adders or multipliers. Based on this spatial granularity, different power-gating schemes that have been proposed in literature are analyzed and several tradeoffs are done. For example the sleep-transistor size dependent leakage reduction is compared and the impact on the state transition wake-up time as well as on the energetic break-even time are analyzed. Furthermore, the impact of the semiconductor technology selection, its process variation, and dynamic parameters such as the supply voltage and temperature is explored. At last, the overall area increase is examined for an example design as a function of a delay constraint.

The remainder of this work is organized as follows. Section 2 summarizes variants of technical implementations for power-gating switches that have been proposed in literature. Section 3 then describes the experimental environment that is used for the analyses followed by efficiency, break-even time, and wake-up time tradeoffs. Section 4 summarizes the results.

2 Related Work

The idea of introducing a third network in series to the pull-up and pull-down network of the complementary metal-oxide semiconductor (CMOS) implementation has been invented by M. Horiguchi, T. Sakata, and K. Itoh in 1993 [3]. Their proposed switched-source impedance circuits have evolved to today's sleep-transistor based power-gating schemes. Figure 2 shows the most typical types of implementation that have been proposed so far. In general, the first three schemes apply p-channel metal-oxide semiconductor (PMOS) gating whereas the last scheme implements n-channel metal-oxide semiconductor (NMOS) gating. In the first scheme, a PMOS transistor that is built in a standard threshold voltage (SVT) process is used for powering down the circuit. The power-gated circuit is typically made of SVT transistors to provide highest possible performance. This kind of power-gating implementation is nearly outdated because high threshold voltage (HVT) sleep transistor (indicated by the thick channel in the second scheme) are used to maintain a higher potential at the virtual rail and thus to push the suppression of leakage currents. Another advantage of HVT sleep transistors is a reduced inherent leakage current of the transistor itself being relevant because of its size. A combination of HVT sleep transistors and SVT devices in the circuit is state-of-the-art in today's realization of power-gated designs.

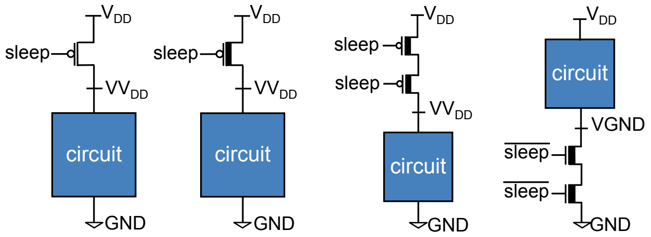


Fig. 2. Microarchitectural implementation of different power-gating schemes

To enforce the leakage suppression, the *sleep* input of a PMOS transistor can be driven by voltage values above V_{DD}. In literature, this technique is referred to as super cutoff CMOS (SCCMOS) [8]. A problem that arises in doing so is a high voltage between the *gate*- and *drain*-terminal of the sleep transistor. This leads to a high voltage stress that may even result in a gate oxide breakdown of the transistor. To overcome this problem the third power-gating scheme applies two sleep transistors in series. A SCCMOS implementation then drives the supply-facing device with a sleep voltage above the supply voltage and the circuit-facing transistor with the supply voltage. Then, a voltage level arises between the two sleep transistors that is uncritical regarding the gate-to-drain voltage of the supply-facing transistor [9]. The transistor stack further leads to a higher voltage drop across the transistors in both, the sleep and active state. This will better suppress the leakage currents during sleep but will also further reduce the virtual supply voltage and thus the speed during operation. Every PMOS gating scheme has a corresponding NMOS-gating based counterpart. For example, the fourth scheme shown in Figure 2 is the NMOS gating counterpart of the one before.

A new field of research is to find alternative power switches like microelectromechanical systems (MEMS) [2] that are physical levers that bend due to any kind of electrostatic, piezoelectric, thermal, or magnetic force and thus form a bonding. In [12], spintronic memristors are proposed for use as power-gating switches. But due to the easier integration into CMOS circuits and the compatibility with today's manufacturing processes, sleep-transistor based power-gating schemes are state-of-the-art and are also addressed in this work.

Beside the general knowledge that $NMOS$ sleep transistors have less impact on the active state and on the state transition due to an inherent higher on-current I_{ON}, no deep tradeoff analysis on the impact of parameters on break-even times, leakage reduction efficiency, and wake-up time has been done so far.

3 Tradeoff Analyses

In the following the power-gating technique is analyzed and the impact of several parameters are traded off. At first, the experimental environment is described. This includes the semiconductor technology selection, and a description of how the necessary estimates are obtained. Then, the impact of semiconductor technology selection on the break-even time is analyzed. Thirdly, the impact of implementation style, supply voltage, temperature, and process variation is analyzed in terms of the power down efficiency and resulting break-even and wake-up times. In the end, the sleep transistor sizing impact and the resulting area increase will further be discussed on an exemplary design.

3.1 Experimental Environment

The proposed power-gating analysis requires consistent models for predicting the power and delay in all operating states. In detail this includes state transition power and delay as well as leakage power models for the power-gated circuit and sleep devices. The overall power-gating estimation framework presenting the E_{SW}, $E_{OVERHEAD}$, I_{SLEEP}, $I_{OVERHEAD}$ is described in [10]. It is completed with the RT-level delay-model of [4] and the I_{ACTIVE} leakage model of [1]. The addressed RT-level components are synthesized to gate-level by Synopsys's Design Compiler for the target technologies and are then converted to Spice-compatible circuits. All used models are then created by HSPICE-based circuit simulations for separately estimating every power state.

The tradeoff analysis in this work bases on an industrial low power $45nm$ technology and on the Nangate free $45nm$ open source digital cell library technology [5]. The latter is a high performance technology based on predictive technology modelcards of the NIMO Group, Arizona State University [6]. It is freely available and is widely used in the scientific context. It further offers three process corners (slow-slow, typical-typical, and fast-fast) and it is an multiple threshold CMOS (MTCMOS) technology. Thus it includes both, standard- and high-V_{TH} devices. All models are characterized for the targeted transistor technology and are temperature and supply voltage dependent.

During the analysis these models are either applied to estimate the effect of power-gating on isolated RT-level components or on multiple components in microarchitectural data-paths that are obtained by behavior- to RT-level synthesis.

3.2 Break-Even Time Analyses of Power-Gating

Figure 3 shows break-even time computations for two different $45nm$ technologies. On the left the steady active and sleep state are characterized in terms of measured I_{ACTIVE} and I_{SLEEP} curves. On the right state transition energy E_{SW} curves are shown. The subsequently break-even time computation (bottom right of the figure) has been performed as defined in Equation 2. All measurements have been performed by Synopsys HSPICE® simulations powering down an 8-bit adder component by PMOS gating with a supply voltage of $1.0V$ and for the two examined transistor technologies. The sleep transistor width W_{ST} remained a parameter as indicated in the charts. Thereby W_{ST} is scaled relatively to the adder component size W_{RT} (defined as the sum of all transistor widths in the pull-up and pull-down networks) in the range of up to 10%.

The results clearly show the t_{be} sensitivity regarding different technologies. While the break-even time of the predictive technology model (PTM) technology ranges between $110 - 122ns$ what is equal to about 12 cycles at a $100Mhz$ clock speed, it is significantly higher for the low power industrial $45nm$ technology. Thus, a temporally fine-grained power down at RT-level only makes sense for the PTM technology whereas the low power technology can only apply power-gating at a system-level perspective for longer idle periods. The same result also holds for the examined $65nm$ technologie in [11] where the authors analyze the break-even time of several circuits to be in the order of one s.

3.3 Power-Gating Efficiency and Dynamic Parameter Impact

This Section will evaluate a system-level view of power management in relative comparisons and absolute numbers against the background of overall possible savings, impact of parameters, and overhead costs of area and power. Thereby, all functional units (FU) in a datapath are power-gated simultaneously.

Table 1 lists design examples and characteristic parameters such as their functional unit datapath composition after high-level synthesis and cycle count within the schedule. To all of the functional units within the designs power-gating has been applied with HVT NMOS sleep devices that are most commonly in todays practice. The fourth and fifth column of Table 1 show absolute I_{ACTIVE} and I_{SLEEP} numbers of the designs at a fixed supply voltage of $1.1V$, an ambient temperature of 27°C, and on the base of the Nangate $45nm$ technology at typical process corner. I_{SLEEP} and I_{ACTIVE} are restricted to the functional units of the designs because of the focus within this work. Nevertheless, the functional units make up the dominating part of the total energy consumption. For example, in

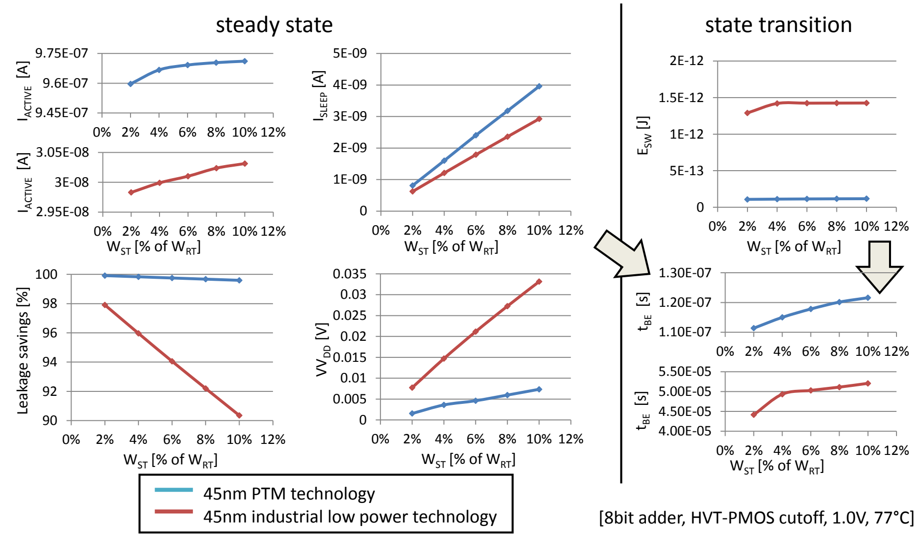


Fig. 3. Break-even time computations of power-gating application

the FDCT benchmark, the functional units contribute 68% of the total energy consumption whereas the remaining 32% split up for multiplexer, registers, controller, and clock tree. As the results show, I_{ACTIVE} is effectively reduced to I_{SLEEP} throughout all benchmarks.

In the following, a deeper analysis of the FDCT benchmark is examined in order to show the impact of the continuous parameters temperature and supply voltage as well as the discrete parameters process corner and power-gating scheme (PGS) selection. For this analysis the HVT version has again been selected for sleep devices and the sleep device sizes have been fixed to 2% for each RTL component. The supply voltage range is constrained to $[1.1V; 1.3V]$ whereas the temperature is examined across a range of [27°C, 127°C]. Figure 4 then shows the gating-switch effectivity as a ratio of I_{SLEEP}/I_{ACTIVE} and the break-even time of the overall FDCT design in nanoseconds.

Table 1. Design examples and the effectivity of power-gating in a global sleep state

Design name	Composition	Schedule length	I_{ACTIVE} of FUs	I_{SLEEP} of FUs
FDCT	4 x add_small@20bit, 3 x sub_small@20bit, 8 x mult_small@20bit	7 cycles	93.1μA	2.2μA
JPEG encoder	1 x add_small@32bit 1 x inc_small@32bit 1 x mult_small@32bit	69 cycles	28.9μA	0.7μA
AES cipher	4 x add_small@32bit 1 x mult_small@32bit	116 cycles	29.9μA	0.7μA

At first, it can be seen that the effectivity of power-gating has only a small variance across the parameter ranges. It becomes only slightly less effective in suppressing leakage currents if the temperature increases. The supply voltage has also only a marginal impact on the effectivity. Additionally, there is only a small variation between 2% and 4% among the different PGSs. In other words, leakage is reduced by 96 – 98% in all cases and, from the point of pure leakage saving, the PGS selection is not particularly interesting if all surrounding parameters are identically.

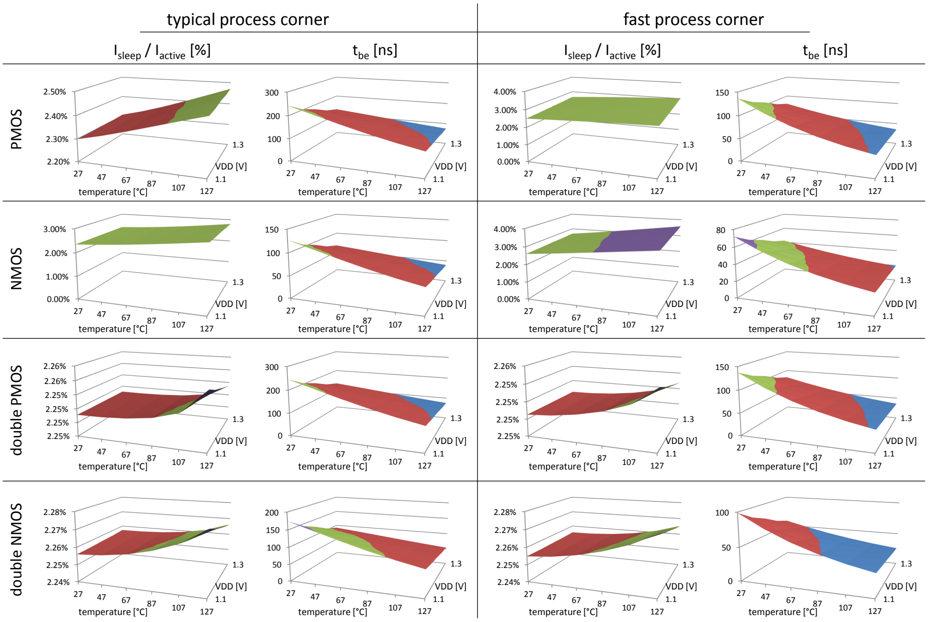


Fig. 4. Comparison of PGS efficiency and dynamic parameter impact

Secondly, the break-even time is presented. Unlike the gating effectivity, t_{be} diminishes with increasing temperature and supply voltage. This is because the wake-up time is much lower and less incomplete transitions occur during the state transition. With a factor of up to four, the variance is also much higher. Furthermore, the PGS selection highly impacts the break-even time. As it can be seen, PMOS schemes have up to two times higher break-even times. Comparing the two process corners, t_{be} is also about twice as big for the typical process corner than that of the fast process corner.

The wake-up time at system-level is given by the maximum RTL component wake-up time if the supply grid is assumed to be sufficiently dimensioned. Figure 5 shows the wake-up time of the FDCT benchmark in dependence on the temperature and supply voltage parameter for the aforementioned gating types and process corners.

It can be observed that t_{wakeup} shows a very small variance in the parameter ranges. It slightly decreases with increasing supply voltage and increases with a

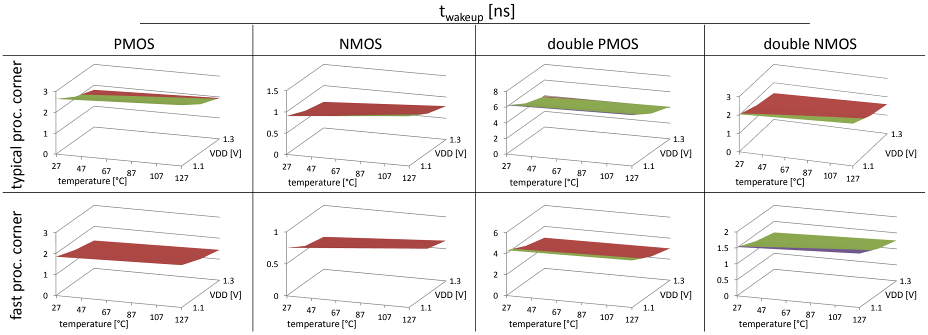


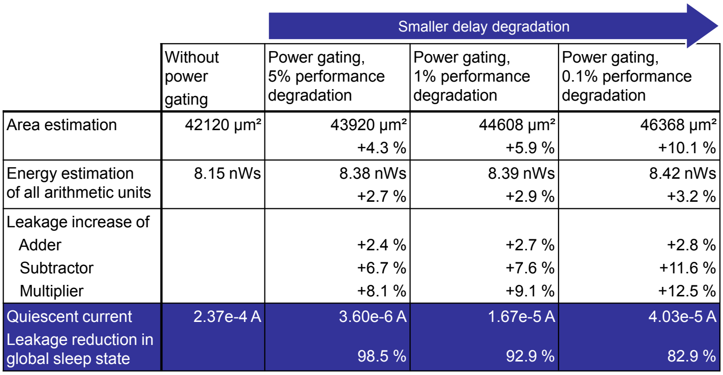
Fig. 5. Wake-up time evaluation of the FDCT design

raising temperature. Furthermore, at the fast process corner, it is about 20 − 30% smaller as it is at the typical process corner. A comparison of the gating schemes shows that NMOS schemes are about three times faster in waking up than PMOS schemes and single-gating schemes are two times as fast as double-gating schemes leading to a total variance of a factor of six.

3.4 Sleep Transistor Sizing Impact

Next, the impact of sleep transistor sizing on the overall power and area demand is analyzed. Therefore, Figure 6 lists four different behavioral synthesis passes of the FDCT behavioral description. In all cases area estimates after rough RT-level placement and energy estimates are provided. Thereby, the area estimates consider sleep transistors, necessary buffers for amplifying the sleep signal, and voltage anchors at the RT-component's outputs. The first column in the table holds the results for a synthesis without power-gating and the other three include power-gating during synthesis process. The difference between them is the maximum allowed performance degradation for each arithmetic unit due to the IR drop accross the sleep transistors during operation. This performance constraint influences the sleep transistor sizing as well as several estimation results including the area, leakage currents during active operation, as well as remaining leakage currents during the global sleep state.

As it can be seen, the area of the design without power-gating is the smallest and the size increases with falling performance degradation. This is because the smaller the allowed delay increase is, the larger the sleep transistor has to be sized to guarantee a worst case delay. In total, the area after rough RT-level floorplanning increases by 4.3% to 10.1% for the overall design. The second row holds the total energy estimation result of all arithmetic units within the design. The pure leakage increase for each component is given in the third row. The overhead that is caused by the additional power management hardware composed of PGS, buffers, and voltage anchors raises up to 3.2%. This increase is due to additional leakage currents of the power-gateable adders, subtracters, and multipliers within the design. For example, an increase in leakage current of up



Smaller delay degradation

	Without power gating	Power gating, 5% performance degradation	Power gating, 1% performance degradation	Power gating, 0.1% performance degradation
Area estimation	42120 µm²	43920 µm² +4.3 %	44608 µm² +5.9 %	46368 µm² +10.1 %
Energy estimation of all arithmetic units	8.15 nWs	8.38 nWs +2.7 %	8.39 nWs +2.9 %	8.42 nWs +3.2 %
Leakage increase of				
Adder		+2.4 %	+2.7 %	+2.8 %
Subtractor		+6.7 %	+7.6 %	+11.6 %
Multiplier		+8.1 %	+9.1 %	+12.5 %
Quiescent current	2.37e-4 A	3.60e-6 A	1.67e-5 A	4.03e-5 A
Leakage reduction in global sleep state		98.5 %	92.9 %	82.9 %

Fig. 6. Power-gating of FDCT design with IP-level granularity

to 12.5% can be observed for a multiplier component compared to a none-power-gateable multiplier. The results show that the more the performance degradation is compensated, the larger the sleep transistors need to be sized leading to an increased area demand and active state energy consumption. Beside the overhead costs, power-gating leads to enormous savings in a global sleep state. The FDCT example shows reductions of the quiescent current of up to 98.5% if a 5% performance degradation is acceptable. In this case, the energy overhead of 2.7% will be amortized by the 98.5% savings if the sleep-time vs. active-time ratio exceeds 3%. If the performance degradation is limited to 0.1%, the overhead increases and the leakage reduction reduces. But even this case is profitable as soon as the entire FDCT design is power-gated for at least 4% of the total time.

4 Conclusion

In this work sleep-transistor based power-gating implementations are analyzed in terms of their efficiency in suppressing leakage currents as well as of their impact on the break-even and wake-up time. The analysis covers typical single/double and PMOS/NMOS-gating implementation schemes.

It is shown that the application area differs a lot and a temporally fine-grained application in terms of powering down for short periods ($< 1\mu s$) is not reasonable in all cases and strongly depends on the semiconductor technology. It is further shown, that the scheme selection as well as the parameters supply voltage, surrounding temperature, and process corner have only a minor impact on the leakage reduction efficiency. In all cases, a leakage reduction of more than 90% has been observed. But the aforementioned parameters highly influence the break-even and wake-up time. Especially the high variance between single- vs. double- gating and the technology process corner stand out.

At any time, the adoption of the power-gating technique is a tradeoff between possible leakage reductions, delay degradation, and area. The latter has been analyzed to typically increase by no more than 10%.

References

1. Helms, D.: Leakage Models for High-Level Power Estimation. PhD thesis, University of Oldenburg, Department for computer science (2009)
2. Henry, M.B., Nazhandali, L.: From transistors to MEMS: throughput-aware power gating in CMOS circuits. In: Proc. of the Design, Automation and Test in Europe Conference and Exhibition 2010 (DATE), pp. 130–135 (2010)
3. Horiguchi, M., Sakata, T., Itoh, K.: Switched-source-impedance CMOS circuit for low standby subthreshold current giga-scale LSI's. IEEE Journal of Solid-State Circuits 28(11), 1131–1135 (1993)
4. Hoyer, M., Helms, D., Nebel, W.: Modelling the Impact of High Level Leakage Optimization Techniques on the Delay of RT-Components. In: Azémard, N., Svensson, L. (eds.) PATMOS 2007. LNCS, vol. 4644, pp. 171–180. Springer, Heidelberg (2007)
5. Nangate Inc. Nangate 45nm open cell library, http://www.nangate.com
6. Nanoscale Integration and Modeling (NIMO) Group at Arizona State University Predictive Technology Model, http://www.eas.asu.edu/~ptm/
7. ITRS Working Group International Technology Roadmap for Semiconductors: Design (2010)
8. Kawaguchi, H., Nose, K., Sakurai, T.: A super cut-off CMOS(SCCMOS) scheme for 0.5-v supply voltage with picoampere stand-by current. IEEE Journal of Solid-State Circuits 35, 1498–1501 (2000)
9. Min, K.-S., Sakurai, T.: Zigzag super cut-off CMOS (ZSCCMOS) scheme with self-saturated virtual power lines for subthreshold-leakage-suppressed sub-1v-vdd LSI's. In: Proc. of the 28th European Solid-State Circuits Conference, pp. 679–682 (2002)
10. Rosinger, S., Helms, D., Nebel, W.: RTL Power Modeling and Estimation of Sleep Transistor Based Power Gating. In: Azémard, N., Svensson, L. (eds.) PATMOS 2007. LNCS, vol. 4644, pp. 278–287. Springer, Heidelberg (2007)
11. Shin, Y., Seomun, J., Choi, K.-M., Sakurai, T.: Power gating: Circuits, design methodologies, and best practice for standard-cell vlsi designs. ACM Transactions on Design Automation of Electronic Systems (TODAES) 15, 1–37 (2010)
12. Wang, X., Chen, Y.: Spintronic memristor devices and application. In: Proc. of the Design, Automation and Test in Europe Conference and Exhibition (DATE), pp. 667–672 (2010)

Modelling and Analysis of Manufacturing Variability Effects from Process to Architectural Level

Chenxi Ni, Ziyad Al Tarawneh, Gordon Russell, and Alex Bystrov

MSD Group, School of EEE, Newcastle University, Newcastle upon Tyne, UK
{Chenxi.ni,Ziyad.al-tarawneh,G.Russell,A.Bystrov}@ncl.ac.uk

Abstract. This paper describes the development of a cell library which can be used to efficiently predict the distribution of circuit delay and leakage power performance due to process variation effects. In developing the library a stepwise approach is adopted in which the effects of process variations on the design parameters of interest at the various levels of design abstraction are evaluated, that is from transistor through circuit to architectural level. A cell library is generated comprising functional blocks whose complexity ranges from a single gate up to several thousand gates. As a demonstration vehicle a 2-stage asynchronous micropipeline is simulated using the cell library to predict the subsequent delay and leakage power distributions. The experimental results show that the proposed method is much faster than the traditional statistical static delay/power analysis (SSTA/SPA) approaches by a factor of 50; the results are also compared with Monte Carlo simulation data for validation purposes, and show an acceptable error rate of within 5%.

1 Introduction

When there is a degree of uncertainty in the fabrication process, the potential effects of the fluctuations implicitly impact not only on device manufacturability and reliability but also on design 'aggressiveness' which affects design performance and subsequently the profitability of the final product.

The statistical variation analysis from process parameters to device performance is commonly based on the Design of Experiment (DoE) technique and Response Surface Methodology (RSM) [2], which enables designers to construct empirical models from which the output responses can be determined as a function of the input factors or parameters. The variability effects from device level to circuit level can be analyzed using Statistical Static Timing Analysis (SSTA) and Statistical Power Analysis (SPA) techniques [3-10], wherein the variation sources are described as Random Variables (RVs) and the performance parameters are usually modelled as low-order polynomials of all the RVs [3].

Although a lot of research into process variation effects has been made at each stage of analysis, there are very few methods which provide a complete methodology to model these effects from the perspective of a designer in terms of circuit performance factors. On the other hand, at present, most of the investigations into statistical

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 11–20, 2013.

delay and leakage power models have been limited to small circuit models. In order to perform an efficient variation analysis on a large circuit, it is essential to take into consideration the effects of process variation on delay and leakage power at a higher level of design abstraction.

This paper introduces a statistical approach based on Technology Computer Aided Design (TCAD) tools and statistical techniques to model the impact of process variation effects on the device performance metrics for devices realised by bulk Silicon process technology. This technique has been applied to the modelling of the variation effects from 16 typical process parameters on the zero-biased threshold voltage, V_{th0}, of the devices. The accuracy of this technique is checked using the 'goodness' of the second-order fit, and the experimental results show that the errors of the mean and sigma value of V_{th0} estimated using the proposed methodology are within 4%. Furthermore, a statistical cell library characterization methodology is proposed that efficiently migrates the effect of process variation on delay and leakage power at device and circuit level to higher levels of abstraction where the overall effect on system performance can be analyzed and design modifications made to ameliorate these effects early in the design cycle. As a demonstration vehicle, the models have been implemented in a 2-stage pipeline circuit. The experimental results show that this methodology can achieve a relatively high accuracy in which the error in the mean and sigma values for delay and leakage power are under 5% compared with 5000-sample Monte Carlo (MC) data, with a computation time which is at least 50 times faster than traditional SSTA/SPA approaches.

2 Analysis of Process Variation Effects on Device Performance

A statistical approach based on TCAD and statistical techniques to model the impact of process variation effects on the device performance metrics for NMOS and PMOS transistors realised by bulk Silicon process technology, is outlined in this section. The general methodology for studying variability is shown in Fig.1 and involves parameter screening, model building and model analysis. The methodology begins with the calibration of the TCAD process and device electrical characteristics with the experimental data, and the extraction of the compact model parameters for nominal devices. During the process simulation, to generate an accurate set of device characteristics, all necessary physical models were incorporated in order to have as realistic a simulation as possible. The basic model for the complete process simulation consisted of a diffusion model with charged point defects, a transient dopant clustering model, a three-phase segregation model and a mechanical stress model, including the thermal and lattice mismatch as well as intrinsic stresses. Thereafter the compact models for NMOS and PMOS transistors were extracted for the nominal devices using I-V data generated.

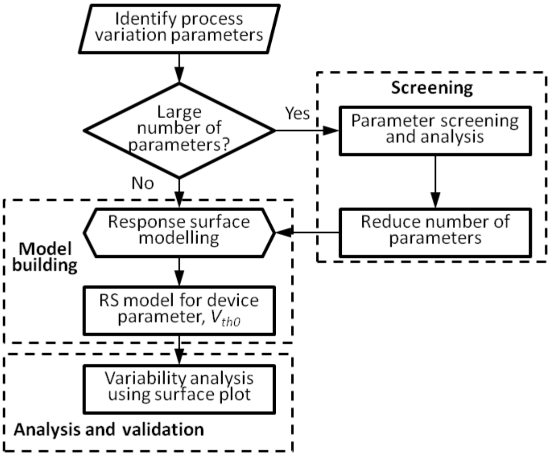

Identify process variation parameters
Large number of parameters?
Yes
No
Screening
Parameter screening and analysis
Reduce number of parameters
Model building
Response surface modelling
RS model for device parameter, V_{th0}
Variability analysis using surface plot
Analysis and validation

Fig. 1. Flow chart of variability analysis utilized by DoE and RSM statistical techniques

The compact model parameter chosen in this work is V_{th0} which represents the threshold voltage for long channel devices at zero bias voltage. The reason for choosing V_{th0} out of the numerous other compact model parameters is that it shows a strong statistical relationship with circuit performance metrics.

3 Analysis of Process Variation Effects on Circuit Performance

In this section, a description is given to the first-order statistical modelling methodology, which will be applied to model the process variation effects on circuit performance in terms of propagation delay time and leakage power dissipation, followed by a general description of the corresponding statistical analysis techniques.

3.1 Statistical Gate Delay and Leakage Power Models

In statistical gate performance modelling, device and circuit environmental parameters will be represented by a Random Variables (RV), which are usually assumed to be Gaussian. The circuit performance parameters are modeled as low-order polynomials of the source RVs. Equations (1) and (2) show the 1^{st} order canonical form for the delay and leakage power models [4][10].

$$D(Delay) = \mu_D + \sum_{i=1}^{n} \beta_{Di} G_i + \beta_{D(n+1)} R \tag{1}$$

$$LP(Leakage\ Power) = exp\left(\mu_P + \sum_{i=1}^{n} \beta_{Pi} G_i + \beta_{P(n+1)} R\right) \tag{2}$$

Equation (1) is the first-order canonical gate delay model. μ_D is the mean delay time of the gate. G_i represents the i^{th} global variational source (Inter-die). R is the sum of all the local RVs in the gate (Intra-die). $\beta_D s$ are the sensitivity coefficients for all the

RVs in this delay model. The gate leakage power on the other hand is modelled as a lognormal RV in Equation (2) since the gate leakage current has an exponential relationship with the variational sources. All the RVs in Equations (1) and (2) follow a normal distribution (Gaussian).

3.2 Statistical Timing and Power Analysis

Both of the timing and leakage power analysis techniques can be used to estimate the overall probability density functions (PDFs) of multiple cell models in terms of delay and power. In order to keep the delay and power analysis alive, both of the techniques use a corresponding approximation methodology to define the non-normal analysis result in a normal canonical form. A small error will be introduced by doing so; however this is not significant (within 5% as shown in the experimental results in section 5). The tightness probability based SSTA approach from C.Visweswariah [4] and the recursive moments-matching based SPA technique from A.Srivastava et al [9] are employed in this work. Having established the cell model form and the analysis methods, the device parameter variation effects can be analyzed on circuit performance in terms of delay and leakage power dissipation. Furthermore, a statistical cell library for 90nm technology has been built to bring the process variation effects up to the architectural level.

4 Analysis of Process Variation Effect at Architectural Level

In order to model process variation effects at architectural level, a statistical cell library comprising a variety of functional blocks has been built. In this section, it will discuss the characterization of the library cells will be discussed in detail.

4.1 Cell Characterization

The gate delay is not only relevant to the device parameters under variation, but also is highly correlated to its operating conditions such as load capacitance C_L and input signal slope T_{in}. Fig.2 shows this relationship between propagation delay and operating conditions for an inverter circuit. It is very difficult to model the operating condition effects (T_{in} and C_L) on propagation delay in canonical form, typically the table look-up approach will solve this problem, where the delay time is sampled with respect to a range of C_L and T_{in} values, then saved in memory. In this work, 7 typical C_L and T_{in} values are sampled as break points, any delay values in close proximity to these will be estimated from the linear function of its adjacent break points. Fig.3 shows a piecewise linear fitting of the inverter delay which is a more efficient view in terms of volume of data required for its representation.

Unlike the statistical delay cells, the characterization of leakage power does not require to be the modelling of the input signal slope because there is no input signal transition in a static environment; furthermore, the leakage power dissipation of a cell has no relationship with its load capacitance, hence there is no need for it to be considered. The leakage power of a cell will only differ when the cell's state has been changed. Hence, only one 2-D lookup table is sufficient to characterize the leakage power of a whole cell where the rows represent the different input signal states, and the columns represent the coefficients in the canonical power polynomial form.

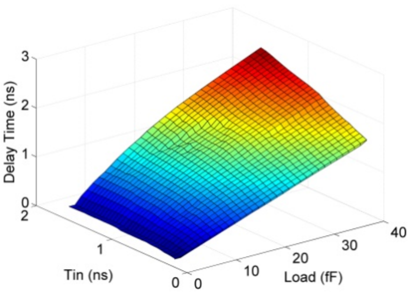


Fig. 2. Inverter delay vs. C_L and T_{in}

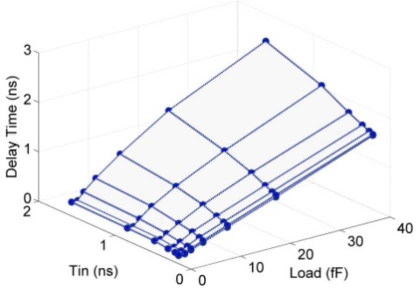


Fig. 3. Modelling using look-up table

4.2 Modelling Higher Level Blocks

By using the methodology introduced in this section, all the basic cells for the logic gates in the library can be created. With the help of the SSTA/SPA techniques, the delay and leakage power performance of any circuit can be analyzed. Consequently, the higher level digital blocks can be modeled using SSTA/SPA analysis results from lower level cells, instead of using SPICE runs; it is very time consuming to run SPICE simulations on larger circuits in order to construct the look-up tables for the delay model. Fig.4 shows a schematic view of variability aware cell modeling framework, which illustrates process variation effects propagating from transistor level to architectural level.

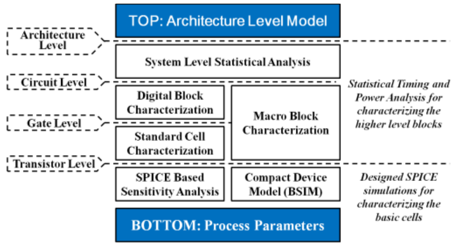

TOP: Architecture Level Model
Architecture Level
System Level Statistical Analysis
Circuit Level
Digital Block Characterization
Macro Block Characterization
Gate Level
Standard Cell Characterization
Statistical Timing and Power Analysis for characterizing the higher level blocks
Transistor Level
SPICE Based Sensitivity Analysis
Compact Device Model (BSIM)
Designed SPICE simulations for characterizing the basic cells
BOTTOM: Process Parameters

Fig. 4. Schematic view of variability aware cell modeling [12]

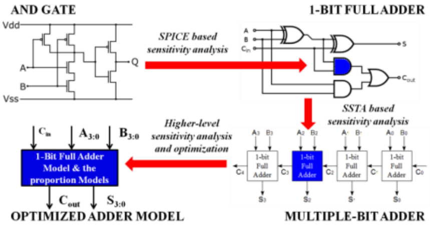

AND GATE
Vdd
A
B
Vss
Q
SPICE based sensitivity analysis
1-BIT FULL ADDER
A
B
Cin
S
Cout
SSTA based sensitivity analysis
Higher-level sensitivity analysis and optimization
Cin
A3:0
B3:0
1-Bit Full Adder Model & the proportion Models
Cout
S3:0
OPTIMIZED ADDER MODEL
1-bit Full Adder
MULTIPLE-BIT ADDER

Fig. 5. Characterization Flow from Standard Cells to 4-bit Adder

Once a digital block has been characterized, it can be used as the standard cell to perform SSTA/SPA at a higher level in a more complex circuit, expending the cell library to architectural level blocks in a hierarchical manner. Since only the variability calibrated results of top level digital blocks are used, the models permits a very fast delay analysis to be performed, which also makes it more suitable for scaling up to a larger system. Fig.5 shows an example of the library block characterization flow from standard cell to a ripple carry adder. The only problem is that a more complex system has a larger number of input transition cases, which leads to a massive memory requirement in order to model each circuit switching case. However, larger functional blocks always have a lot of symmetry and multiple occurrences in the circuit. In Fig.5, the ripple carry adder is actually a serial connection of multiple full adders, so it can be characterized just by the full adder model. During the work of constructing the whole library, most of the blocks can be represented using a smaller circuit model, the output delay time is simply a matter of the proportions of the model.

5 Implementation and Experimental Results

To study and analyze the impact of process variability on the compact model parameter V_{th0}, sixteen process parameters were identified as potential sources of uncontrollable variation at different process steps for the devices implemented in a bulk silicon technology. All the process parameters were varied by ±10% of their mean values. However the process temperatures during the different manufacturing steps were set at ±10 ^{o}C from the nominal. This is because the temperature values are very high and, in practice, would not drift in the range of ±10%. The range of variation (± 10% and ±10 ^{o}C) corresponds to ±3σ variation. The Pareto plots indicating the relative magnitude of the effects which various process parameters have on V_{th0}, for NMOS and PMOS devices are shown in Fig.6-7.

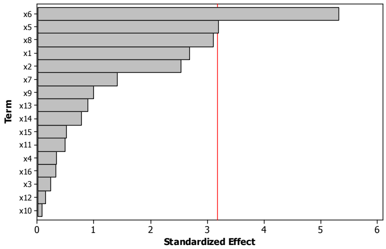


Fig. 6. Pareto plots for NMOS device

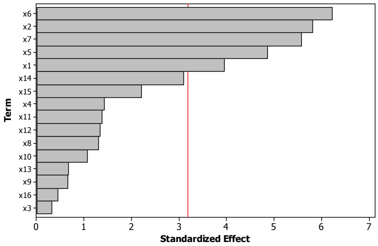


Fig. 7. Pareto plots for PMOS device

Table 1 summarizes the most significant process steps that influence V_{th0} for both NMOS and PMOS devices based on the observation from the Pareto plots.

Table 1. Most significant parameters that impact V_{th0} for NMOS and PMOS devices

NMOS	PMOS
V_{th} adjustment implantation energy (x_6). V_{th} adjustment implantation dose (x_5). Halo implantation energy (x_8). Gate oxide thickness (x_1). High-k dielectric thickness (x_2).	V_{th} adjustment implantation energy (x_6). High-k dielectric thickness (x_2). Halo implantation dose (x_7). V_{th} adjustment implantation dose (x_5).

Having identified the most significant process parameters for the device responses in the screening steps in terms of the compact model parameter, namely V_{th0}, the response surface models for these compact model parameters were subsequently built. Fig.8 and Fig.9 show the response surface for V_{th0} for the NMOS device as a function of different process parameters.

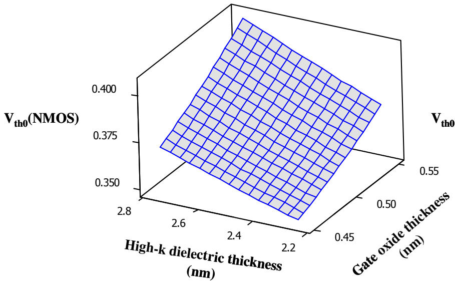


Fig. 8. Response surface for V_{th0} (NMOS) with respect to gate-oxide thickness and high-k dielectric thickness

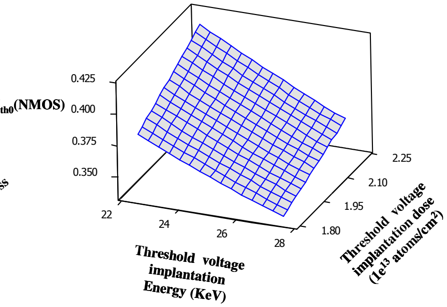


Fig. 9. Response surface for V_{th0} (NMOS) with respect to implantation energy and dose

Table 2. The variation result of V_{th0} for both NMOS and PMOS devices

Device Type	V_{th0}(V)				
	μ	σ	σ /u	R2	R2adj
NMOS	0.3975 v	0.0362 v	10.98%	97.93%	96.05%
PMOS	-0.3957 v	0.049 v	12.38%	98.38%	96.10%

Table 2 compares the μ and σ values of V_{th0} for both NMOS and PMOS devices followed by the 'goodness' of the second-order fit. It can be seen that the *R2* and *R2adj* values are very close to 100% for all responses. This is desirable and therefore ensures that the models accurately highlight the variability due to process fluctuations.

Since the mean and sigma values have been established and verified as accurate, V_{th0} can be assumed as a variation source and applied to the cell models in the statistical cell library which was introduced in Section 4. For demonstration purpose, V_{th0} of the NMOS and PMOS devices are assumed to be normal-distributed local variables with sigma value equal to 11% and 12% of its mean values respectively; additionally the supply voltage V_{dd} and operating temperature T are chosen as the global variation sources, which are also normal variables, which vary ±15% of their mean values. (The mean value of V_{dd} and T is 1v and 75 °C respectively)

Subsequently, the process variation effects were analyzed on a 2-stage asynchronous pipeline circuit using the cell library with the assumed parameter specification above. The pipeline circuit is an event-controlled two-phase bounded data system [11] with an overall complexity of 3272 gates which is shown in Fig.10. It contains 2 asynchronous pipeline registers, a 3-to-8 decoder, a 16-bit register file with 16 memory cells and a

16-bit ALU which can perform 8 logic and arithmetic functions. All the circuit cells were realized using BSIM 4.0 90nm technology model card from University of California, and simulated using Berkeley SPICE program. The cell library is implemented in Matlab Simulink, which provides a convenient graphic interface of all cell blocks to the user. All the transistor parameters in one gate or basic cell are assumed to have a high correlation to each other, and share the same local variables.

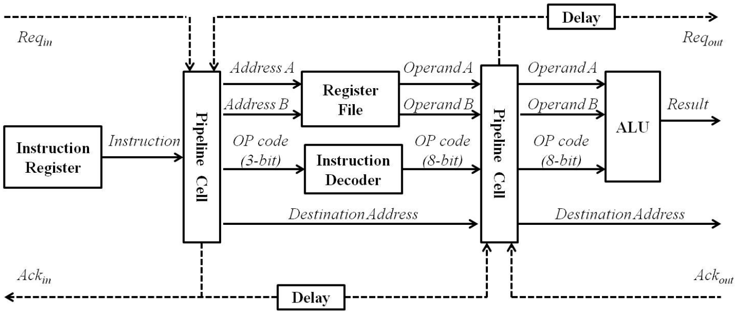


Fig. 10. Pipeline processor block diagram

The pipeline circuit has also been simulated using SPICE-base Monte Carlo technique for validation purposes. As an example, Fig.11 and 12 show the PDFs for the propagation delay time and leakage power dissipation of the ALU block. The histograms in Fig.11 and 12 are generated by a 5000-sampled Monte Carlo simulation, and the solid lines are the predicted PDF obtained using the cell library. It can be seen from the graphs, the predicted PDF fit the Monte Carlo data very well.

All the experimental results are compared with 5000 sampled Monte Carlo simulation, the errors in the delay and leakage power distributions for the main blocks used in the pipeline circuit are listed in Table 3. All the errors are within 5%, 3% for most of the cases. Furthermore, the proposed cell library is also much more feasible compared with MC simulation in analyzing the process variation effects on circuit performance; it would take a month to run a 5000 sampled MC simulation for the pipeline circuit in Fig.10 whereas the cell library only requires a few seconds. The speed-up factors shown in Table 3 have illustrated this advantage of the proposed technique.

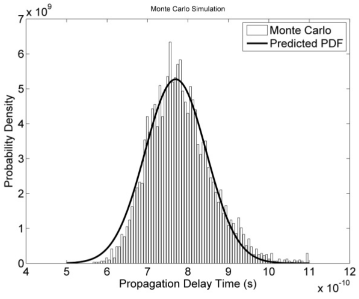


Fig. 11. Delay PDF Matching for ALU

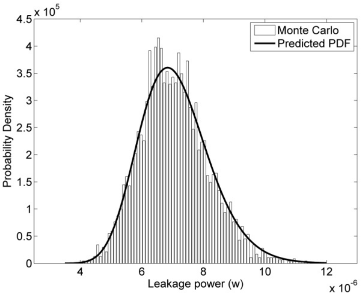


Fig. 12. Leakage PDF matching for ALU

Additionally Table 3 also compares the CPU time between the cell library and traditional SSTA and SPA approaches; it shows that using the cell library is also much faster in computing the delay and leakage power distribution of circuits over traditional SSTA and SPA approaches. The speed-up factor is highly related to the regularity of the circuits. For example, the computation time for the performance analysis of the Register File (RF) block is at least 100 times faster than SSTA and SPA; this is because there are a large number of identical digital blocks (registers) in the circuit of the RF which can be represented by a single model block. On the other hand the speed-up factor for the decoder block is only around 10 since most of the decoder circuit is modeled at gate level. The experimental results show that the overall speed-up factor for the whole demonstration pipeline circuit is more than 50.

Table 3. Comparison of Results

Blocks	**No. of Gates**	**Compared with 5000 Sampled MC Simulations**					**Computation Time Comparison**	
		Delay Error		**Leakage Error**		**Speed-up Factor**		
		μ (%)	σ (%)	μ (%)	σ (%)		**Cell Library**	**SSTA & SPA**
Decoder	*21*	*1.26*	*4.65*	*1.76*	*1.89*	*9,543*	*0.84s*	*11.8 s*
ALU	*743*	*0.62*	*2.32*	*0.73*	*3.45*	*20,246*	*2.40s*	*33.9 s*
Register File	*2574*	*0.79*	*1.90*	*1.87*	*3.49*	*28,965*	*0.96s*	*98.7 s*
Pipeline	*3738*	*1.14*	*1.88*	*3.11*	*4.33*	*58,209*	*3.27s*	*166.1 s*

6 Conclusions

A statistical methodology based on the DoE and RSM techniques with the aid of TCAD, has been utilized to study and analyze the effect of process variability in a 90nm bulk silicon technology; a similar approach can be used for SOI technologies, FINFET structures etc. and also for the more advanced technology nodes when process variation issues will be more problematic. Subsequently, a methodology for the architectural level modelling of the effects of process variation on propagation delay and leakage power was introduced. A statistical cell library has been built in order to provide both speed and efficiency in analyzing circuit delay performance. All the desired device parameters, whose variation specifications are obtained from the DoE and RSM analysis, are assumed to be Gaussian variables. A first-order canonical delay model is employed. The cell model can not only precisely reflect the process variation effect on delay and leakage power, but also can cope with different operating conditions and switching cases. This makes it easier to construct higher level blocks in the library with the help of SSTA and SPA. Based on the proposed methodology, the variation effects from manufacturing process can be analyzed at architectural level. A full analysis has been demonstrated on a 2-stage micropipeline circuit;

where it has been shown that this technique can achieve a comparable accuracy compared to a Monte Carlo simulation with the errors less than 5%, and save significant amount of computation time, at least 50 times faster than traditional SSTA and SPA approaches.

It is considered that the technique outlined in this paper, permits designers to efficiently assess the effects of variations in processing parameters, such as threshold voltage, together with supply voltage and operating temperature, on a design in terms of their potential impact on specification parameters such as propagation delay and leakage power early in the design cycle. Subsequently, the designer can choose which technology or cell library should be used to implement the design to ensure its robustness to the effects of process variation.

References

1. Kang, S., Leblebici, Y.: CMOS Digital Integrated Circuits, 2nd edn. Mc-Graw Hill (1999)
2. Montgomery, Design and Analysis of Experiments, 5th edn. John Wiley and Sons (2001)
3. Okada, K., Yamaoka, K., Onodera, H.: A Statistical Gate-Delay Model Considering Intra-Gate Variability. In: ICCAD, pp. 908–913 (2003)
4. Visweswariah, C., Ravindran, K., Kalafala, K., Walker, S.G., Narayan, S.: First-order incremental clock-based statistical timing analysis. In: DAC 2004, pp. 2170–2180 (2004)
5. Zhan, Y., Strojwas, A., Li, X., Pileggi, L., Newmark, D., Sharma, M.: Correlation aware statistical timing analysis with non-Gaussian delay distribution. In: DAC 2005 (2005)
6. Bhardwaj, S., Ghanta, P., Vrudhula, S.: A framework for statistical timing analysis using non-linear delay and slew models. In: ICCAD 2006, pp. 225–230 (2006)
7. Singhee, A., Singhal, S., Rutenbar, R.A.: Practical, Fast Monte Carlo Statistical Static Timing Analysis: Why and How. In: ICCAD 2008, pp. 190–195 (2008)
8. Rao, R., Srivastava, A., Blaauw, D., Sylvester, D.: Statistical Estimation of Leakage Current Considering Inter- and Intra-die Process Variation. In: International Symposium on Low Power Eletronics and Design, pp. 84–89. ACM, New York (2003)
9. Srivastava, A., Kaviraj, C., Shah, S., Sylvester, D., Blaauw, D.: A Novel Approach to Perform Gate-Level Yield Analysis and Optimization Considering Correlated Variations in Power and Performance. IEEE Trans. Computer-Aided Design 27(2), 272–285 (2008)
10. Chang, H.: Full-chip analysis of leakage power under process variations, including spatial correations. In: Proc. ACM/IEEE DAC 2005, pp. 523–528 (2005)
11. Ivan, E.: Sutherland “Micropipelines”. Communications of ACM 32(6), 720–738 (1989)
12. Dierickx, B., Miranda, M., et al.: Propagating Variability from Technology to Architectural level. In: Workshop on IWPSD 2007, pp. 74–79 (2007)

Non-invasive Power Simulation at System-Level with SystemC

Daniel Lorenz[1], Philipp A. Hartmann[1], Kim Grüttner[1], and Wolfgang Nebel[2]

[1] OFFIS – Institute for Information Technology,
Escherweg 2,
26121 Oldenburg, Germany
{lorenz,hartmann,gruettner}@offis.de

[2] Carl von Ossietzky Universität,
Ammerländer Heerstr. 114-118,
26121 Oldenburg, Germany
nebel@informatik.uni-oldenburg.de

Abstract. Due to the increasing algorithmic complexity of today's embedded systems, consideration of extra-functional properties becomes more important. Extra-functional properties like timing, power consumption, and temperature need to be validated against given requirements on all abstraction levels. For timing and power consumption at RT- and gate-level several techniques are available, but there is still a lack of methods and tools for power estimation and analyses at system and higher levels. In this paper we present an approach for non-invasive augmentation of functional SystemC$^{\text{TM}}$ TLM-2.0 components with power properties. The I/O behaviour of a TLM-2.0 component will be observed by a Protocol State Machine (PrSM) that generates trigger events to stimulate a Power State Machines (PSM). The PSM describes the component's internal power states and transitions and transitions between them. Each component's PSM is connected with a frequency and voltage dependent power model. We present first evaluation results of different IP components and compare our system-level power traces generation with state-of-the-art gate-level power simulations in terms of accuracy and simulation speed.

Keywords: power state machine, system level, energy consumption.

1 Introduction

While the complexity of today's applications as well as the performance is continuous increasing, the development of battery capacities cannot keep up with the power demand of applications. This leads – especially in mobile systems – to reduced system runtimes, if suitable energy saving methods are not used. But entire systems are already very complex requiring power management techniques to be integrated in early design phases, long before a physical hardware platform is available. For this reason it is necessary to be able to analyse the expected

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 21–31, 2013.

power consumption of all system components for realistic system use-cases, e. g. in system-level simulation runs.

This includes consideration of energy consumption of the processor cores (and its integrated power management capabilities) plus the energy consumption of additional hardware components on the platform (e. g. buses, memories, dedicated hardware accelerators). Most of them are not developed from scratch, but reused from previous designs or bought from IP vendors. Usually these IP components do not contain any information about their power consumption. In some cases data sheets for calculating an estimate of the power consumption in dependence of the clock frequency, supply voltage, and target technology are available. But information of these data sheets can hardly be used in functional system-level simulations, because they are not applicable to any dynamic effects. These include different platform usage scenarios, which bring along different load profiles per platform component and thus different energy consumption per component over time.

To simulate complex applications on today's platforms, abstract and high performance simulation models are necessary. Today's state-of-the-art Virtual Hardware Platforms enable early software development. Component models in these platforms make use of Transaction-Level Modelling (TLM) techniques [1]. For this purpose the SystemC™ TLM-2.0 Standard[2] allows description and simulation of inter-component communication using memory mapped I/O (e. g. on a shared bus).

To enable power management of the entire system on high abstraction levels, a holistic approach for energy estimation based on virtual platforms in SystemC TLM-2.0 is required. This paper presents a concept for non-invasive simulation of energy consumption for SystemC TLM-2.0 system-level components. The term "non-invasive" describes that our proposed power annotation does not require any modifications on the functional TLM models, nor on the SystemC simulation kernel. The internal state and energy consumption of TLM components is approximated on the observation and interpretation of the interaction with the environment. This property allows us to use existing black-box IP components and commercial simulation environments that do not allow modification of the SystemC simulation kernel.

Section 2 gives an overview about existing related work for modelling energy consumption in SystemC. Our approach for describing and simulating *Power State Machines* (PSM) is presented in Section 3. With the aid of a PSM the energy consumption (resp. the average switched capacitance) for every abstract internal state of an IP component is defined. To detect the transitions between these states, the examined component's communication behaviour is observed at its interfaces through a specific adapter called *Protocol State Machine* (PrSM). In Section 4 the energy consumption of different IP components is modelled and simulated at system-level. The resulting power traces are compared against energy simulations on gate-level. Section 5 closes with a summary and gives an outlook on future work.

2 Related Work

In the past, several methods for estimating the energy consumption on different abstraction levels have been investigated. It became evident that early design decisions offer the greatest potential on power savings [3]. In [4] Power State Machines have been proposed for the first time in system-level modelling. State transitions are controlled by inputs of a power manager. Instead of a definite energy consumption, a state can have a maximal energy consumption, too. Its activity level is controlled by a power manager. Costs for state transitions are modelled by delays.

Since dynamic energy consumption emerges from activity, in [3] a concept is presented where activity is observed at the communication interfaces. A process is notified about every `default_event` of the communication channels and traces the activity. This method is very transparent and produces little overhead, but neglects the activity inside the component which can definitely behave different than the activity in the communication channels.

With *Powersim* a new approach was recently proposed in [5]. In this case the model does not have to be changed because a modified SystemC simulation kernel is used. The power model of the system is loaded at the beginning of the simulation, which does an activity estimation at every access on a SystemC data type. Due to the modified SystemC simulation kernel this approach cannot be integrated in commercial design environments, which have a proprietary SystemC kernel that usually cannot be changed. Additionally an estimation of communication, especially of abstract models (e. g. TLM), is possible only with restrictions because merely the assignment operator can be used for power annotation.

The past has shown that modelling and estimating the energy consumption through state machines with adaption to the environment is a good possibility. In [6] this approach is used with SystemC/TLM to invoke a state transition in a state machine by calling a method at corresponding source positions in the functional model. Energy consumption can be adapted to Dynamic Voltage and Frequency Scaling (DVFS). The disadvantage of this approach is its invasiveness and thus can hardly be used for IP components. Its use with black-box IP components is impossible.

In [7] a system-level model that separates architecture and functional application model is presented. The application model is mapped on the platform model, which includes a model for the energy consumption. By using configuration files, different mappings of the application model onto the platform model can be explored. In contrast to our approach the application is an actor-based Dynamic Data Flow (DDF) Model of Computation (MoC). This approach assumes an application model described by a DDF MoC. Our proposed approach does not make any restrictions to the used MoC. It can be used with any IP component that communicates via memory-mapped I/O. For this reason our approach could be used at the platform model in [7].

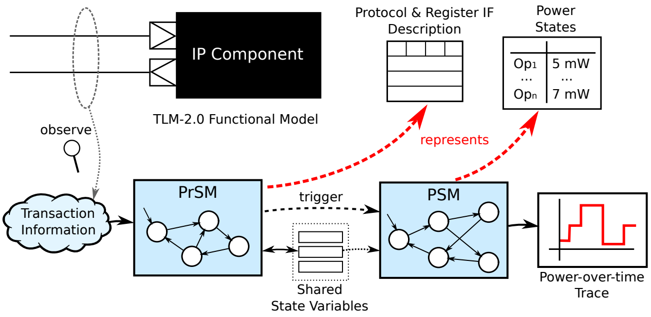


Fig. 1. PSM approach overview for non-invasive simulation of energy consumption

3 Power State Machines in SystemC

An essential requirement for the proposed approach is the estimation of black-box components. In this case the internal structure cannot be annotated or modified to record switching activities to compute energy consumption. For this reason the internal state of an IP component is abstracted based on a correlation of its observable interaction with the estimated power consumption. The estimated energy consumption per macro states can be taken from data sheets, can be inferred from an estimation of design size or the functional complexity of the design (top-down), or can be extracted from lower level (e. g. RTL) simulations (bottom-up). With the latter approach, the most accurate model can be built, c. f. [6].

Fig. 1 gives an overview of our non-invasive approach for integration of energy information with TLM IP components. For using our approach the following is assumed:

- The examined IP component is either available as synthesisable functional model with TLM-2.0 interface or as a functional black-box (TLM) model and synthesisable black-box RT model.
- The register interface, i. e. the logical register organization in the global address space, and its usage has been defined.
- In addition to the various operation modes of the component an energy estimation exists (average power consumption at a given target technology or the component needs to be synthesizable for generating a power-over-time trace).

Based on the energy characteristics of the component the dynamic power model can be created. To determine the dynamic energy consumption $P(t)$ as function of time, the well-known formula

$$P(t) = \frac{1}{2}V_{\mathrm{dd}}^2 fC \cdot \alpha(t) \tag{1}$$

can be used, where C is the total switched capacitance which is always constant, V_{dd} the supply voltage, f the clock frequency, and $\alpha(t) = \{x \mid 0 \leq x \leq 1\}$ the switching activity at time t.

Since the real switching activity $\alpha(t)$ cannot be obtained at system-level, a set of macro states $S = s_1, \ldots, s_n$ is defined in the PSM to distinguish characteristic operation modes. These are time dependent, thus the time $t \in T$ can be associated with the operation modes $s \in S$ as follows:

$$\varphi : T \rightarrow S \tag{2}$$

We assume that time is already quantified into a discrete set of values. To each operation mode $s_i \in S$ an average activity, described by the arithmetic mean $\alpha_i \in A$ for the time, when s_i is active, is associated in the following way:

$$\alpha' : S \rightarrow A \quad \text{ with } \alpha_i = \frac{1}{\Delta t(s_i)} \sum_{t=t_{\text{beg}}(s_i)}^{t_{\text{end}}(s_i)} \alpha(t) \tag{3}$$

Inserted in (1) we get the average energy consumption $\overline{P}(t)$ over time:

$$\overline{P}(t) = \frac{1}{2} V_{\text{dd}}^2 f C \cdot \alpha'(\varphi(t)) \tag{4}$$

Supply voltage and frequency are parameters depending on the power island and the current strategy of the power manager. For this reason we abstract from these parameters and get the average switched capacitance over time:

$$\overline{C}(t) = \frac{1}{2} C \cdot \alpha'(\varphi(t)) \tag{5}$$

In (5) the output of the PSM is described. This formula is the basis for recording the average switched capacity during simulation to generate a power-over-time trace.

Since the used components are modelled with TLM-2.0, the known external register-based interfaces of a component are observed over time to approximate the internal functionality. Based on these observations, a *Protocol State Machine* (PrSM) is controlled. The main task of the PrSM is the triggering of state transitions in the PSM through observation and interpretation of the interaction between component and environment. The PrSM extracts the energetic relevant events to orthogonalize the communication and functional artefacts of the non-functional PSM model. This can lead to a reduction of complexity in the PSM because it only describes the different internal operation modes whereas the PrSM covers the protocol state of the component. Furthermore, the separation of PrSM and PSM has the advantage that components with the same access protocol and different internal implementations could use the same PrSM, only the PSM has to be changed.

The input data for the PrSM is the observed transaction information of the TLM-2.0 sockets of the component. Depending on the register interface and the semantics of read/write operations at specific address, activities or even complex operations are triggered in the observed component. This can be modelled with transitions between states in the PrSM to abstract the functional characteristics. Such transitions can lead to notifications of the PSM.

Furthermore, there may be some functional state transitions in the IP component that cannot be detected by observing the interaction with the environment (e. g. if a target component finishes its calculation, it may not send a transaction indicating this, because target components do not initiate transactions at all). In this case the PSM is modelled as a state machine that executes state a transition after a defined delay/timeout.

It is possible to extend PrSM and PSM with shared state variables Z. These shared state variables are modified only by the PrSM and can be used to store specific configuration data, like current function choice (e. g. MD5 vs. SHA256 in a hashing module). This modification facilitates a more easy management of context aware information to be shared between PrSM and PSM. With this modification the output of the PSM also depends on the shared state variables and the determination of the average switching activity from (4) and (5) has to be modified to $\alpha'(\varphi(t), Z(t))$. I.e. state transitions, delays, and output of the state machines can depend on the shared state variables. If a shared state variable describes the current clock frequency of the module, delays in the PSM can be adapted accordingly.

3.1 Protocol Observation

Under the assumption that the examined component interacts with the environment via TLM-2.0 *Base Protocol* [2], several information with the following attributes are available:

- Generic Payload (R/W, base address, payload data, data length, . . .)
- Current phase of transaction (TLM-2.0 LT/AT base protocol phase)
- Used interface of component (socket)
- Role of the component in the transaction (initiator, target, or interconnect)

To model the PrSM, the communication with the component can be observed at multiple protocol layers. On one hand this is the mere read/write operation in the address space of a component (*transport layer*, single TLM-2.0 transaction). For a memory component this layer is mostly sufficient to describe the energetic behaviour because there is no logic relation between multiple transactions.

On the other hand for complex components, like hardware accelerators, side effects inside the target component can occur over several transactions (*presentation layer*). In this case it is an advantage to separate the address space of the component in control, configuration, and payload addresses. Therefore, the semantic description of the register interface is needed to abstract the functionality with the aid of suitable states and shared state variables in a PrSM.

Control data influence the protocol flow and thus immediately the internal state of the component. An example for this is an explicit handshake register to synchronize with a hardware accelerator. Transactions at such registers ordinary

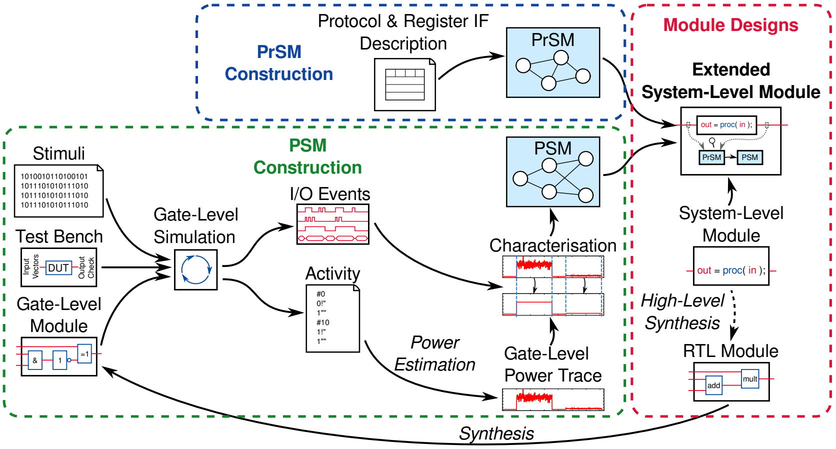


Fig. 2. Flow for creating extended system-level modules from system-level modules

lead to explicit state transitions in the PrSM. *Configuration data* influence the behaviour of a component in the following states or over a long period of time (e. g. the choice of the algorithm for an encryption coprocessor). The values of configuration registers often require the introduction and observation of a state variable in the PrSM. *Payload data* are processed by the IP component, but they do not influence the communication protocol or the internal (observable) protocol flow of the component (e. g. data that should be encrypted by the above mentioned accelerator). Although the concrete used data influences the activity in a component, this is in most cases very small and can be neglected on system-level (see experiments in Section 4).

3.2 PSM Model Construction

For construction of the PSM model we assume plateaus in the power-over-time trace in which the average energy consumption keeps nearly constant over time. These plateaus are described through power states in the PSM model. That means, if a state transition in the PSM occurs, there is a change in the average energy consumption, i. e. a change between plateaus.

To generate these power values from gate-level simulations, a test bench is used that implements a use case in which preferably all operation modes are tested. During simulation the activity inside the device under test is traced and converted into a power-over-time-trace with the aid of target technology and timing information. Such a trace is shown in Section 4. This trace, overlayed with I/O events over time, provides information about periodic behaviour of energy consumption and communication dependent and independent switches of plateaus. Based on this analysis relevant power states, the average switched capacitance per power state, and transitions between power states can be extracted. Additionally, special communication attributes can be associated with

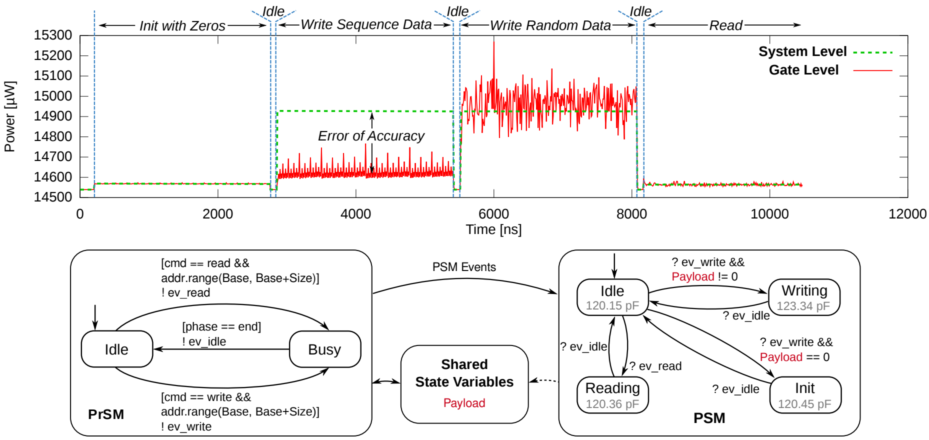


Fig. 3. Comparison of gate-level and system-level power-over-time trace of a Synopsys® DesignWare® SRAM IP with related PrSM and PSM

power state transitions. This includes state transitions with annotated switching delays that cannot be triggered through I/O events. The complete flow for creating the PrSM and PSM models is sketched in Fig. 2.

4 Experiments

To evaluate the proposed approach, we applied the PSM method to a Synopsys® DesignWare® SRAM IP and an AES encryption module. We synthesized the RT modules to gate-level and generated power-over-time traces to detect the different operation modes, corresponding I/O operations, and calculated the average energy consumption for every state. With this information the PSMs and PrSMs have been implemented and new stimuli data have been applied to the gate-level and system-level simulation model. We compared the execution times for simulating and calculating the average energy consumption, and the accuracy of the average energy consumption between system-level and gate-level simulation models.

The SRAM model is quite simple and can be divided into four operation modes: idle, initialise, writing and reading. These modes can also be seen in the PSM in Fig. 3. To detect the initialise state, the payload data is saved in the shared state variable which can be accessed by the PSM. Fig. 3 shows also the gate-level power-over-time trace overlayed with the system-level power-over-time trace (dotted line) for a test bench with the following phases: initialisation with zeros, writing a sequence of numbers, writing random data, and reading back data. It can be seen that initialising with zeros consumes only a small amount of energy and has been considered for modelling in a separate state (Init). Writing a sequence of numbers consumes also much fewer energy than an MP3 stream, because in average only two bits are switching between two

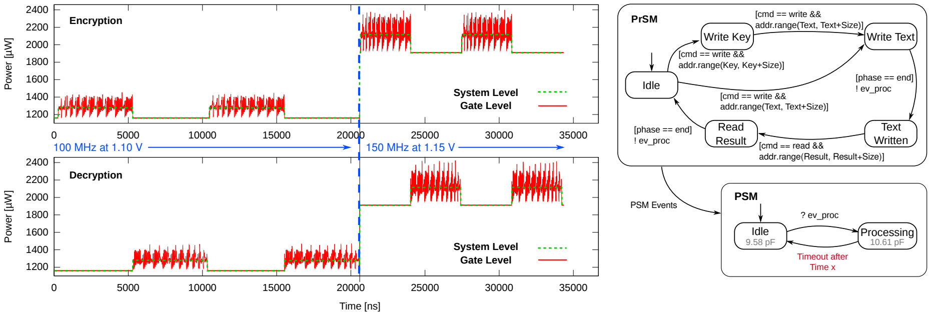


Fig. 4. Comparison of gate-level and system-level power-over-time trace of an AES encryption and decryption module

payload data. Nevertheless, the worst-case error of average energy consumption measured between the system-level and gate-level power-over-time traces was 2.44% when initializing the SRAM with zeros. For this reason it was put in a seperate state to reduce the error to 2.05%. For a typical set of examined stimuli data, we have measured an average execution time of about 7 ms at system-level and 13.225 s at gate-level.

For the AES module we took two different implementations in SystemC. The first one is at RT-level and cycle accurate. The second one is modelled at transaction-level. For the transaction-level model and RT-level model we took the same power values and delays and therefore the power-over-time traces of these models are equal. The data is first encrypted (setting the AES module to encryption mode) and decrypted afterwards by a second AES module that works in decryption mode. In this example the PSM uses the timeout mechanism to detect the end of the encryption/decryption process because no communication takes place after the AES processing has been finished. During simulation a Dynamic Frequency and Voltage Scaling (DFVS) from 100 MHz at 1.10 V to 150 MHz at 1.15 V has been done. A comparison of system-level and the gate-level power trace is shown in Fig. 4. To generate the gate-level power trace it took 3.5 hours. For this model we also measured a worst-case error for the average power consumption of below 1%, because for a crypto core the energy consumption does not significantly depend on the input data or on the operation mode (encryption/decryption). We measured that the execution time at RT-level in SystemC is about 6.66 faster compared to gate-level simulation. At transaction-level the execution time is even 3800 times faster than at gate-level, it takes only 15 ms.

In our experiments the static power had an amount of about 7% of the total energy consumption and the variation accounted only for 10%. For this reason we assumed a constant static power.

These first experiments show very promising results. Modelling the energy consumption at system-level with PSMs can lead to a significant speed-up > 1000, while offering a sufficiently accurate power-over-time tracing at the same time. The only drawback resides in the relatively high modelling effort for generating the PrSM and PSM model for the first time. Moreover, the RTL power-over-time trace generation also take some time, since state-of-the-art gate-level power estimation tools only support the calculation of average energy consumption over a specific time frame. Hence, each sample has to be calculated separately and concatenated to a power-over-time trace before the PSM model can be generated. But this effort only has to be spent once per IP component.

5 Conclusion and Future Work

In this paper we have presented a methodology for non-invasive modelling of the energy consumption of transaction-level models. Through observation at the interfaces and abstract interpretation of the internal power states, modifications of the functional model or the underlying simulation kernel can be avoided. The PrSM is the communication protocol interpreter and filters power information to trigger PSM transitions. This way also introspection of transmitted payload data can be filtered to trigger power mode switches. Through modelling the PSM as state machine extended with timeouts also time dependent and communication independent power mode switches can be represented. We have applied our approach to an SRAM and AES encryption IP and obtained a significant simulation speed-up in comparison to gate-level simulations. We could also confirm a high fidelity of the system-level power simulations. In the future we will apply this methodology to components with a more complex internal state space like a DMA controller and examine the correlations between I/O operations and gate level power trace. Additionally we want to take into account parasitic effects.

Acknowledgements. This work has been partially supported by the EU integrated project COMPLEX under grant FP7-247999 and by the BMBF project EnerSave under grant 01BE1102.

References

1. Cai, L., Gajski, D.: Transaction level modeling: an overview. In: Proceedings of the 1st IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS 2003, pp. 19–24. ACM, New York (2003)
2. IEEE Computer Society: IEEE Standard SystemC Language Reference Manual. IEEE Std 1666–2011 (2011) ISBN 978-0-7381-6802-9
3. Walravens, C., Vanderperren, Y., Dehaene, W.: ActivaSC: A highly efficient and non-intrusive extension for activity-based analysis of SystemC models. In: 46th ACM/IEEE Design Automation Conference (DAC 2009), pp. 172–177 (July 2009)
4. Benini, L., Hodgson, R., Siegel, P.: System-level power estimation and optimization. In: International Symposium on Low Power Electronics and Design, ISLPED 1998, pp. 173–178. ACM, Monterey (1998)

5. Giammarini, M., Orcioni, S., Conti, M.: Powersim: Power Estimation with SystemC. In: Conti, M., et al. (eds.) Solutions on Embedded Systems. LNEE, vol. 81, pp. 285–300. Springer Science+Business Media, B.V. (2011)
6. Lebreton, H., Vivet, P.: Power Modeling in SystemC at Transaction Level, Application to a DVFS Architecture. In: IEEE Annual Symposium on VLSI, ISVLSI 2008, pp. 463–466. IEEE Computer Society (April 2008)
7. Streubühr, M., Rosales, R., Hasholzner, R., Haubelt, C., Teich, J.: ESL Power and Performance Estimation for Heterogeneous MPSoCs Using SystemC. In: Forum on specification and Design Languages (FDL 2011), Oldenburg, Germany, pp. 202–209. IEEE Computer Society (September 2011)

A Standard Cell Optimization Method for Near-Threshold Voltage Operations

Masahiro Kondo[1], Shinichi Nishizawa[1],
Tohru Ishihara[1], and Hidetoshi Onodera[1,2]

[1] Graduate School of Informatics, Kyoto University,
[2] JST, CREST, Japan

Abstract. Near-threshold voltage operation is a well-known solution for drastically improving the energy efficiency of microprocessors fabricated with the latest process technologies. However, it is not well studied how the optimal gate size of standard cells changes when the supply voltage of the microprocessors gets closer to the threshold voltage. This paper first shows an experimental observation that the optimal gate size for near-threshold voltage which is 0.6V in this work is far from the optimal gate size for the nominal supply voltage which is 1.2V in our target process technology. Based on this fact, the paper next presents our cell optimization flow which finds the optimal gate sizes of individual standard cells operating at the near-threshold voltage. The experimental results show that, when operating at the 0.6V condition, the energy consumptions of several benchmark circuits synthesized with our standard cells optimized for the 0.6V condition can be reduced by 31% at the best case and by 23% on average compared with those of the same circuits synthesized with the cells optimized for the nominal supply voltage.

1 Introduction

With an increasing number of portable electronic devices and with a severe world-wide energy crisis, there is a great demand for designing low energy LSI circuits. Although many low energy design techniques have been developed before and they have contributed for reducing the energy consumption in the LSI circuits, ever increasing demand for higher-performance LSI circuits requires reducing more energy in the LSI circuits. Since the energy consumption of the LSI circuits is quadratically proportional to the supply voltage, reducing the supply voltage of the LSI circuit is very effective for the energy reduction. However, it increases delay in the LSI circuit. Therefore, it is necessary to take the energy-delay trade-off into consideration for designing the high-performance and low-energy LSI systems. Standard cells are key components for designing high quality LSI circuits in a short turn-around-time. The standard cells are a set of pre-designed and pre-verified building blocks which can be used for automatically synthesizing a targeting circuit. Since the energy consumption and the performance of the standard cells strongly affect those of the target circuit synthesized with the standard cells, it is customary that a cell library is designed

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 32–41, 2013.

carefully by a silicon foundry such that the library can fully exploit the characteristics of its target fabrication process. Typical standard cells are tuned for a nominal operating voltage of the targeting process technology. In the latest process technology, 1.2V or 1.0V is the nominal voltage. Therefore, if we use the cells in lower voltage conditions like 0.6V or 0.5V, several design parameters of the cells are not necessarily optimal. This paper shows an experimental observation that the optimal gate size for near-threshold voltage operation is much smaller than that for the nominal supply voltage. Based on this observation, we find the optimal gate width for each cell in a target standard cell library which reduces in balance the energy consumption and the delay of the circuits synthesized with the cells. Although there are several previous studies on the optimization of PMOS/NMOS size ratio [3,4], to the best of our knowledge, our work is the first one which explicitly shows that the optimal total size of the PMOS and the NMOS in a cell also depends on the target supply voltage. The rest of this paper is organized as follows. Section 2 describes related work and motivation of this work. Section 3 shows our cell optimization flow and results of individual cells. Section 4 shows energy consumption, area and performance results of benchmark circuits synthesized with our optimized cells and foundry provided cells, respectively. Section 5 concludes this paper.

2 Related Work

In recent years, standard cells targeting for near- and sub-threshold voltages are widely investigated. In [1,2], cell libraries optimized for a wide range of supply voltage are presented. Their main focus is designing standard cells covering wide range of operating voltage instead of optimizing the cells for a specific low voltage. Although this optimization policy is very effective for designing energy efficient DVFS-capable processors which run across a wide supply voltage range in order to support both high performance and high power efficiency modes of operation, it is not necessarily the best solution for the processors if they target a specific low operating voltage. Unlike their approaches, our approach optimizes the cells for a specific near-threshold voltage. Our optimization policy is good for multi-VDD designs where each voltage island is optimized for a specific operating voltage. This paper shows the effectiveness of our approach by comparing the energy efficiency of several benchmark circuits synthesized using our cells optimized for a near-threshold voltage with that of the same circuits synthesized using foundry-provided cells. In [3,4], methods for optimizing the P/N width ratio of standard cells for a specific low supply voltage are proposed. Although the methods in [3,4] find the optimal P/N width ratio, they do not try to find the optimal gate size (i.e., the total size of PMOS and NMOS transistors). Unlike [3,4], our approach simultaneously finds the optimal gate size and the optimal P/N ratio of cells for the near-threshold voltage operation. Above mentioned previous papers also do not quantitatively show the impact of the cell optimization on the energy consumption, area, and performance of practically large circuits in near- and sub-threshold conditions. Even if individual cells are very energy efficient, the large circuits synthesized with the cells are not always very energy efficient. This is because the quality of the circuits synthesized with a cell library strongly depends on a holistic quality of the cells in the library. Therefore, evaluating the large scale circuits synthesized by the cells is very important. This paper quantitatively shows the impact of the cell optimization on the energy consumption, area and performance of practically large scale benchmark circuits selected from ISCAS'85 benchmark suite. In [5,6], methods for choosing the gate size and the supply voltage of the circuit simultaneously at a circuit synthesis phase are proposed. Those papers show that the wider gate saves more energy consumption of the circuits synthesized with the gates when the supply voltage is scaled down together. This is because the supply voltage quadratically impacts on the energy consumption of the circuit while the gate size only linearly affects the energy consumption. Therefore, increasing the gate size together with lowering the supply voltage can reduce the energy consumption without any delay overhead. This is true when we chose the gate size and the supply voltage at a circuit synthesis phase. However, when we develop a standard cell library, the cells are optimized for a specific supply voltage such that the library cells can fully exploit the characteristics of their target fabrication process at the specific supply voltage condition. In this paper, for the first time, we show that the optimal gate size for a near-threshold voltage operation is smaller than that for a nominal supply voltage operation.

3 Standard Cell Optimization

This section presents our cell optimization method which finds the optimal gate sizes of individual cells in a target standard cell library. The target process used in this work is a commercial 65nm CMOS process technology. The optimization is done by changing the gate sizes of foundry-provided standard cells. Therefore, the basic architecture, the height and the placement pitch of our standard cells are the same from those of the foundry-provided cells.

3.1 Target Cells

We target standard cells presented in Table 1. Although the target cells do not include a full set of the foundry-provided cells, the full variety of drive strength

Table 1. Type of cells in a target library

Logic	Drive Strength
INV	0.5X, 1X, 1.5X, 2X, 3X, 4X, 6X, 8X, 10X, 12X, 16X
BUF	0.5X, 1X, 2X, 3X, 4X, 6X, 8X, 10X, 12X, 16X
NAND2, NOR2	0.5X, 1X, 1.5X, 2X, 3X, 4X, 6X, 8X
AND2, OR2	0.5X, 1X, 2X, 3X, 4X, 8X
AOI21, OAI21	0.5X, 1X, 2X, 4X
AOI22, OAI22	0.5X, 1X, 2X

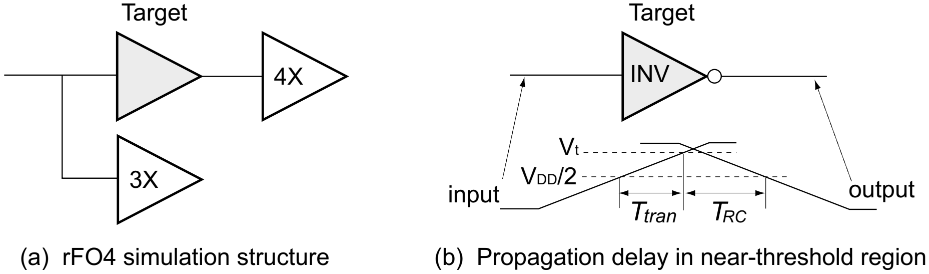
values is included in the target library. Functions of AOI21, OAI21, AOI22, and OAI22 are as follows;

$$\begin{aligned} AOI21 &= !((A \cdot B) + C) & AOI22 &= !((A \cdot B) + (C \cdot D)) \\ OAI21 &= !((A + B) \cdot C) & OAI22 &= !((A + B) \cdot (C + D)) \end{aligned}$$

The total number of cells included in the target library is 63.

3.2 Simulation Setup

In a typical simulation structure for finding the optimal gate size, rFO4 is used [3] as shown in the left of Figure 1. An FO4 (fanout of 4) is used for both input and output. By keeping the reference input and output cells unchanged, various gate sizes of the target cell are compared [3]. This structure works well when the supply voltage is enough high compared to the threshold voltage. However, in a near-threshold voltage region, this structure does not necessarily find the optimal gate size for the final circuits synthesized with the cells. As shown in the right of Figure 1, the propagation delay can be defined as a difference between times where the input and the output voltages become $V_{DD}/2$, respectively. In the near-threshold region, the transition delay of input signal (T_{tran} in Figure 1) is dominant since the threshold voltage (V_t in Figure 1) is higher than the $V_{DD}/2$. Therefore, if the gate size of the reference input is kept unchanged, a smaller gate size in the target cell results in a smaller propagation delay of the target cell. This does not reflect an actual situation of circuit synthesis using the standard cells. For example, when we synthesize a circuit, the gate sizes of all the cells in a standard cell library used for the synthesis are pre-optimized for a specific supply voltage. In this case, the gate size of the input cell is small if the gate size of the target cell is small. To reflect this situation more accurately, we use a new FO4 simulation structure where only the output load of the target cell is kept unchanged for finding the optimal gate size of the target cell as shown in Figure 2. In our gate size optimization, the optimal gate size is exhaustively explored by changing the gate size of all the cells except for the FO4 cells connected to the output of the target cell under evaluation.



(a) rFO4 simulation structure

(b) Propagation delay in near-threshold region

Fig. 1. rFO4 simulation structure

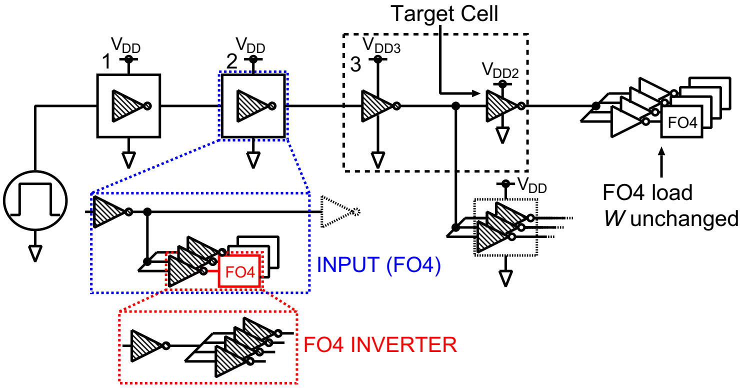


Fig. 2. New FO4 simulation structure

3.3 Optimization Flow

A flow of our gate size optimization is shown in Figure 3. For a given cell in a target library, the propagation delay D and the energy consumption E for various gate widths are measured through circuit simulation using HSPICE of SYNOPSYS. From the results, we calculate an energy and delay square product (ED^2) value for each cell. Note that we measure rise (D_{rise}) and fall (D_{fall}) propagation delays separately. The ED^2 value here is obtained by $E \times (D_{rise}^2 + D_{fall}^2)$. Then, we find the optimal NMOS and PMOS gate sizes separately for each cell, which minimize the ED^2. The gate size exploration begins from the minimum gate size allowed in the target process technology. Then the gate size is increased by Δ_{w} at a time, and E and D values are evaluated for the gate size. This process is repeated until the improvement of the D value in the target cell becomes negligible. From all the gate sizes explored in this procedure, we find the optimal gate size which minimizes the ED^2 value. Finally, based on the gate sizes of the cells, we find a common WELL boundary among cells so that the total cell area is minimized. If the gate width is larger than the WELL

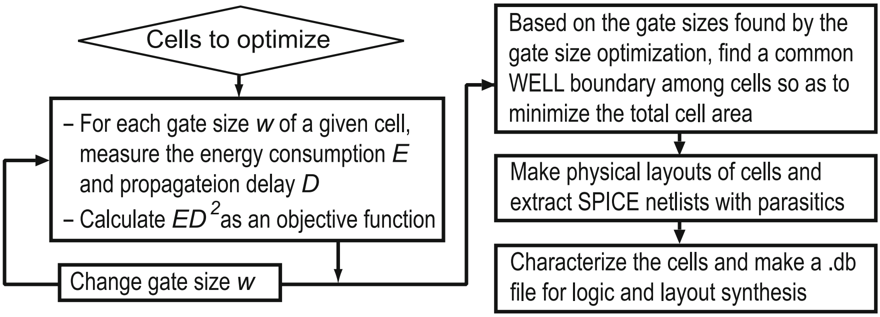

Cells to optimize
– For each gate size w of a given cell, measure the energy consumption E and propagateion delay D
– Calculate ED 2 as an objective function
Change gate size w
Based on the gate sizes found by the gate size optimization, find a common WELL boundary among cells so as to minimize the total cell area
Make physical layouts of cells and extract SPICE netlists with parasitics
Characterize the cells and make a .db file for logic and layout synthesis

Fig. 3. Optimization flow

height, the cell is implemented as a multi-finger cell. Physical layouts of cells are created based on the results of gate size and WELL boundary optimizations. We use NCX of Liberty for characterizing the cells. The characterization results are compiled into a .db file which is used for logic and layout synthesis.

In [7], validity of ED^x is discussed. The x represents an exponent for delay. Values of x less than one mean that the energy is more significant than delay, and on the other hand, values larger than one can be interpreted as a case that the delay restriction is more severe than the energy one. In many circuit optimization flows, a metric of ED is used. This metric treats both energy and delay factors in a same way. However, minimizing ED product without considering the application does not give us any reasonable answer. Suppose we reduce both the energy and delay by tuning the gate size in the near-threshold region. As explained in the previous subsection, since the transition delay is dominant in the near-threshold region, reducing the delay by increasing the gate size is more difficult than reducing the energy by decreasing the gate size. Therefore, using more than one for x in the ED^x metric seems reasonable. In this work, we therefore use an ED^2 metric instead of using the ED product.

3.4 Optimization Results

The gate size optimization results for the near-threshold voltage condition are summarized in Figure 4. Upper two figures show gate width ratios of our cells to the foundry-provided cells. Lower two figures show the ED^2 ratios of our cells to the foundry-provided cells. The most important observation here is that the optimal gate size for the near-threshold voltage condition is much smaller than that for the nominal supply voltage. The main reason of this result is that the cells in the near-threshold operation have a weaker dependence between the gate width and the delay compared to that in nominal voltage operation. As explained with Figure 1 in Subsection 3.1, the input transition delay is dominant in the total propagation delay when we use the cells in a near-threshold voltage region. Therefore, the propagation delay for the near-threshold operation is less sensitive to the gate width compared to that for the nominal voltage operation. On the other hand, the relation between the energy consumption and the gate width is almost independent from the operating voltage. As a result, for the near-threshold voltage operation, the optimal gate width which minimizes the ED^2 of the cell is smaller than that for the nominal voltage operation. More specifically, on average, the gate size of the cells optimized for the near-threshold voltage is 22% smaller than that of the foundry-provided cells. As a result, the energy consumption of the cells optimized for the near-threshold voltage is reduced by 22% on average compared to the foundry-provided cells as well when we use them in the near-threshold region.

However, as we mentioned above, even if individual cells are very energy efficient, the large circuits synthesized with the cells are not always very energy efficient. This is because the quality of the circuits synthesized with a cell library strongly depends on a holistic quality of the cells in the library. Therefore, evaluating the large scale circuits synthesized with the cells is very important. In the

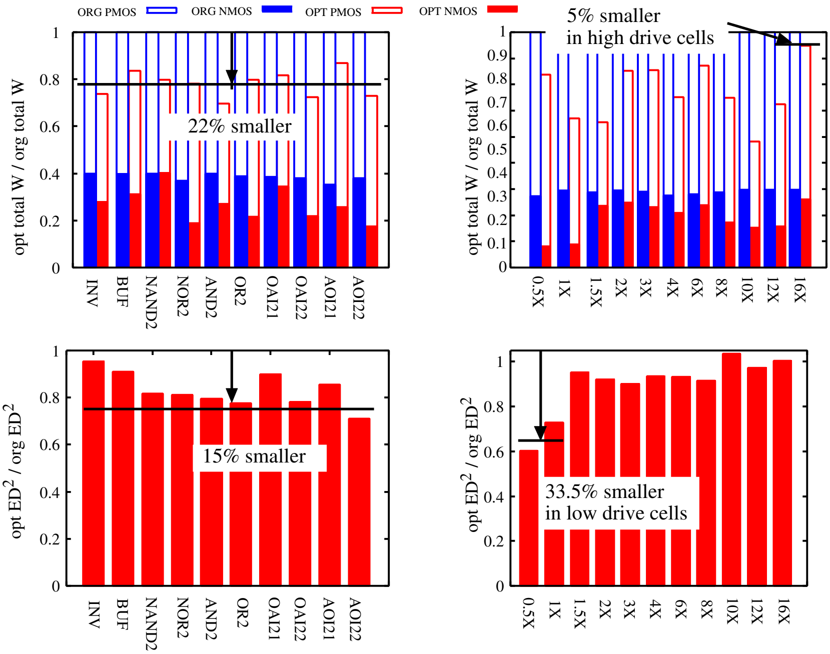


Fig. 4. Gate size optimization results for near-threshold operation

next section, we quantitatively shows the impact of the cell optimization on the energy consumption, area and performance of practically large scale benchmark circuits selected from ISCAS'85 benchmark suite.

4 Evaluation with Benchmark Circuits

4.1 Experimental Setup

We use four different benchmark circuits selected from ISCAS'85 benchmark suite as summarized in Table 2. We use *design_compiler* of SYNOPSYS for synthesizing the circuits.

4.2 Logic Synthesis Results

Figure 5 shows the energy consumption of the benchmark circuits for different time constraints in near-threshold voltage operation. Solid and dotted lines in the figure show the results for benchmark circuits synthesized with foundry-provided standard cells and our standard cells optimized for the near-threshold voltage, respectively. Vertical axis represents the energy consumption of the circuits. Horizontal axis represents the time constraints given when the circuits are

Table 2. Specification of the ISCAS-85 benchmark circuits[8]

Circuit	Function	No. of inputs	No. of outputs	No. of logic gates
c2670	12-bit ALU and controller	233	140	1193
c3540	8-bit ALU	50	22	1669
c5315	9-bit ALU	178	123	2307
c7552	32-bit adder/ comparator	207	108	3512

synthesized using *design_compiler*. The time constraint corresponds to the maximum operating clock frequency of the circuit. The energy consumption values in the figure are estimated by *power_compiler* of SYNOPSYS. As can be seen from the figure, our standard cells can reduce the energy consumption of the circuits drastically. For example in the c7552 circuit, our library reduces the energy consumption by more than 28% compared with the foundry-provided library when 5200ps is given as a time constraint. The circuits synthesized with our cells can also largely improve the maximum circuit performance. More specifically, the maximum clock frequency of the c7552 circuit synthesized with our library is improved by 4% compared with that synthesized with the foundry-provided library. Area of the circuits synthesized with the most strict time constraints are also shown in Figure 5. As can be seen from the figure, area of the circuit syn-

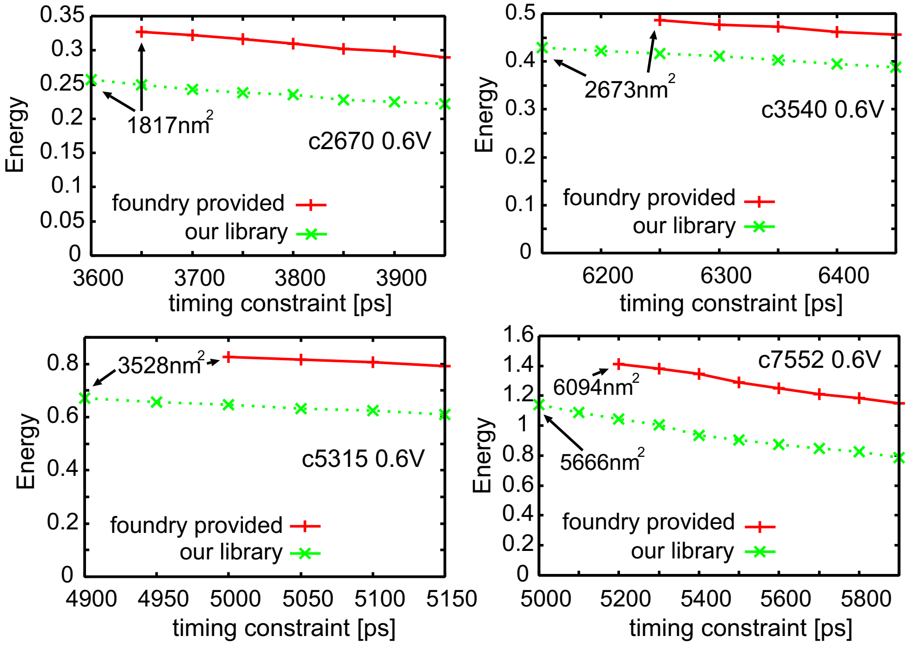


Fig. 5. Energy and performance results for near-threshold voltage operation

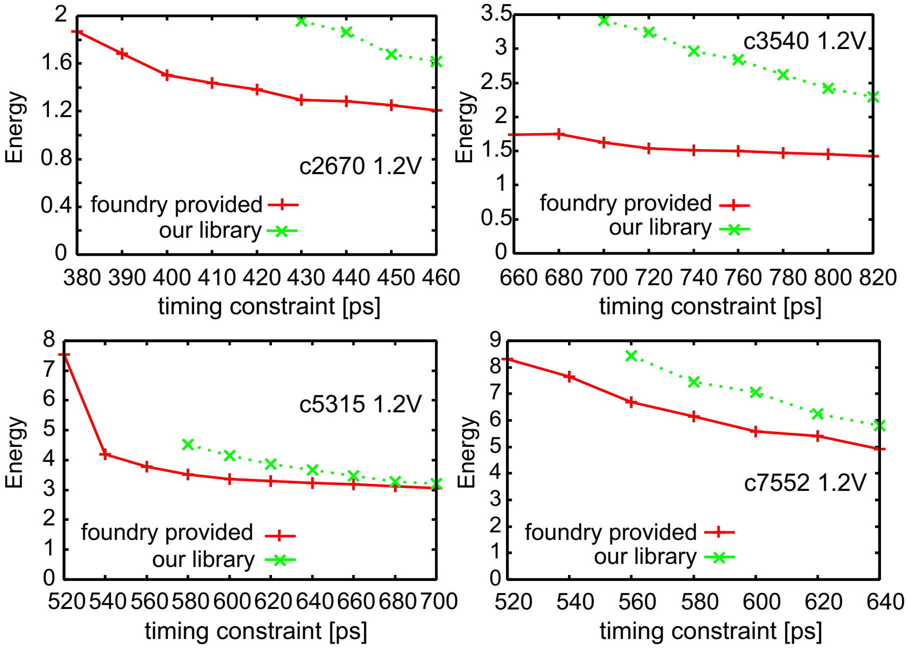


Fig. 6. Energy and performance results for nominal supply voltage (1.2V)

tesized with our library is equal to or smaller than that synthesized with the foundry-provided library.

Similarly, Figure 6 shows the results for 1.2V operation. For obtaining these results, we first characterize both our library and foundry-provided library under the nominal voltage (i.e., 1.2V) condition. The benchmark circuits are then synthesized with those libraries. As can be seen from the figure, on average, the energy efficiency of the circuits synthesized with the foundry-provided cells is better than that of the same circuits synthesized with our cells optimized for 0.6V operation. Another important observation from the figure is that the maximum performance of the circuit synthesized with the foundry-provided cells is higher than that of the same circuit synthesized with our library. This means that the foundry-provided cells can synthesize a high performance circuit which cannot be synthesized by our cells in the nominal voltage condition. In several cases, the energy consumption of the circuit synthesized with the foundry provided cells is very close to that of the same circuit synthesized with our cells. However, this happens only when the time constraints are not very strict. The main reason is that our library has more cells that have low-drive strength than the foundry-provided library. Therefore, when the time constraint is not very strict, those low-drive cells are effectively used for non-critical paths, which reduces the energy consumption of the circuit. From the results shown in Figure 6 and Figure 5, we can conclude that optimizing the gate sizes of the standard cells for a specific target supply voltage has a strong impact on both energy reduction and performance improvement of the circuits synthesized using the standard cells.

5 Conclusion

This paper shows an experimental observation that the optimal gate size for near-threshold voltage (0.6V) operation is much smaller than that for the nominal supply voltage (1.2V). Then, we show that the main reason of this observation is a weaker dependence between the gate width and the delay of the cell in lower voltage conditions. Based on this fact, we find the optimal gate width for each cell in a target standard cell library which reduces in balance the energy consumption and the delay of the circuits synthesized with the cells. The synthesis results show that the energy consumptions of the circuits synthesized with our optimized standard cells can be reduced by 31% at the best case and by 23% on average compared with those of the same circuits synthesized with the foundry-provided cells. Our future work will be devoted to develop a framework which takes the process variation into consideration during the gate size is optimized.

Acknowledgment. This work is supported by the chip fabrication program of VDEC, the University of Tokyo in collaboration with eShuttle Inc. and Fujitsu Semiconductor Ltd. The authors would like to thank JSPS NEXT program for their support to this work.

References

1. Ickes, N., et al.: A 28nm 0.6V Low-Power DSP for Mobile Applications. IEEE JSSC 47(1), 35–46 (2012)
2. Jain, S., et al.: A 280mV-to-1.2V Wide-Operating-Range IA-32 Processor in 32nm CMOS. In: Proc. ISSCC, pp. 66–68 (February 2012)
3. Abouzeid, F., et al.: 40nm CMOS 0.35V-Optimized Standard Cell Library for Ultra-Low Power Applications. ACM TODAES 16(3), Article 35 (June 2011)
4. Kung, D.S., et al.: Optimal P/N Width Ratio Selection for Standard Cell Libraries. In: Proc. ICCAD, pp. 178–184 (November 1999)
5. Hanson, S., Sylvester, D., Blaauw, D.: A New Technique for Jointly Optimizing Gate Sizing and Supply Voltage in Ultra-Low Energy Circuits. In: Proc. of ISLPED, pp. 338–341 (October 2006)
6. Bol, D., Ambroise, R., Flandre, D., Legat, J.-D.: Impact of Technology Scaling on Digital Subthreshold Circuits. In: Proc. of ISVLSI, pp. 179–184 (April 2008)
7. Emadi, M., Jafargholi, A., Moghadam, H.S., Nayebi, M.M.: Optimum Supply and Threshold Voltages and Transistor Sizing Effect on Low Power SOI Circuit Design. In: Proc. of APCCAS, pp. 1394–1398 (December 2006)
8. Hansen, M.C., Yalcin, H., Hayes, J.P.: Unveiling the ISCAS-85 benchmarks: A Case Study in Reverse Engineering. IEEE Design & Test of Computers 16(3), 72–80 (1999)

An Extended Metastability Simulation Method for Synchronizer Characterization

Salomon Beer and Ran Ginosar

Electrical Engineering Department,
Technion – Israel Institute of Technology, 38200 Haifa, Israel
{sbeer@tx,ran@ee}.technion.ac.il

Abstract. Synchronizers play a key role in multi-clock domain systems on chip. Designing reliable synchronizers requires estimating and evaluating synchronizer parameters $\boldsymbol{\tau}$ (resolution time constant) and $\boldsymbol{T_W}$ (metastability window). Typically, evaluation of these parameters has been done by empirical rules of thumb or simple circuit simulations to ensure that the synchronizer MTBF is sufficiently long. This paper shows that those rules of thumb and some common simulation method are unable to predict correct synchronizer parameters in deep sub-micron technologies. We propose an extended simulation method to estimate synchronizer characteristics more reliably and compare the results obtained with other state-of-the-art simulation methods and with measurements of a 65nm LP CMOS test-chip.

Keywords: Synchronization, metastability, synchronizers, simulation, MTBF.

1 Introduction

Multiple-clock domain System on Chip (SoC) designs require synchronization when transferring signals and data among clock domains and when receiving asynchronous inputs. Such synchronizations are susceptible to metastability effects which can cause malfunction in a receiving circuit. In critical designs, this risk must be mitigated. To assess the risk and to design reliable synchronizers, models describing the failure mechanisms for latches and flip-flops have been developed [1][2]. Most models express the risk of not resolving metastability in terms of the mean-time-between-failures (MTBF) of the circuit, Eq. (1), where S is the time allotted for resolution, F_C and F_D are the receiver and sender clock frequencies, respectively, τ is the resolution time constant, and T_W is a parameter related to the effective setup-and-hold time window during which the synchronizer is vulnerable to metastability.

$$\mathrm{MTBF} = \frac{e^{S/\tau}}{\mathrm{T_W} \times \mathrm{F_C} \times \mathrm{F_D}} \tag{1}$$

Over the years, techniques have been developed for obtaining an arbitrarily long MTBF. These techniques have been translated into convenient rules of thumb for designers. As digital circuits have become more complex, denser and faster with

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 42–51, 2013.

reduced power consumption, the old rules of thumb are beginning to fail [3][4], especially when adding process variations and operating-condition sensitivities in today's manufacturing technologies [5]. One rule of thumb has stated that the time constant τ is proportional to the fan-out of four (FO4) propagation delay. This rule of thumb thus predicts that τ decreases with feature size and FO4 gate delay. However, a change in this pattern is emerging at process nodes 90nm and below [3][4][6]. This change is particularly significant when the metastable voltage (typically about ½V_{DD}) is in the vicinity of the transistor threshold voltage, an increasingly common occurrence for low-power circuits employing lower supply voltage and high threshold transistors. Under these circumstances, the current flowing in a metastable complementary pair of transistors can be exceedingly small [4], resulting in a large value of τ. Operating conditions, particularly at low temperatures, and process variations further aggravate the situation and can cause many orders of magnitude variation in the MTBF of a synchronizer. No longer can the designer depend upon the rule of thumb that τ is proportional to the FO4 delay. As a result, traditional guidelines for synchronizer design are no longer useful and simulations should be used to correctly estimate synchronizer error probabilities.

Over the years, several simulation methods have been proposed to calculate synchronizer failure probabilities. In some works [2][3][7], the simulation shorts latch nodes to force metastability to estimate τ. In this work we show that this simple node-shorting method is inadequate for simulating general latches and is only valid for fully symmetric cross-coupled inverters. In non-symmetric latches the method generates incorrect results. Typically, a chain of latches or flip-flops is used for synchronization. Those latches are usually non-symmetric or the capacitive loading by other circuits leads to non-symmetric circuits, yielding inaccurate results. We propose an extension of the simulation method for the case of asymmetric cross-coupled inverters and compare the results of our extended method to results obtained by two other state-of-the-art simulation methods [8][10]. We show that our proposed novel simulation method correctly predicts synchronizer parameters in a simpler manner and at a lower computational cost than another recently published method [8]. To further validate our method we also compare the result of our simulations with real measurements of 65nm LP CMOS latches.

2 Node Shorting Simulation

Node shorting simulation (NSS), as described in [2][3][7], is widely used by designers. The two nodes of a latch are shorted to equate their voltage, simulating metastability. When the short is opened, the latch is allowed to resolve. One node will diverge to V_{DD} while the other to ground. A small battery (order on nV) across the nodes is placed to ensure the starting time and the direction of divergence. Fig. 1 shows two different latch configurations with the voltage controlled switch that is used to short the lacth nodes (dotted lines). The potential metastable nodes a, b are higlighted in red.

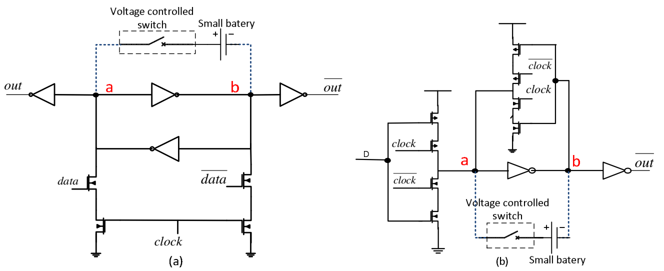


Fig. 1. Two common latch configurations, (a) Reduced clock swing latch [12] (b) regular low high latch configurations [13]

The small signal behavior of the latch can be modeled as two cross coupled inverters as shown in [7][10] and it is possible to describe their behavior by

$$\begin{aligned}\tau_a \dot{v}_a(t) &= -(v_b(t) - v_{ma})\\ \tau_b \dot{v}_b(t) &= -(v_a(t) - v_{mb})\end{aligned} \tag{2}$$

where the time constant $\tau_i = C_i/g_{mi}$, v_{mi} is the metastability voltage at the input of the i^{th} inverter(in the metastable node), C_i is the total capacitance associated with the i^{th} metastable node, g_{mi} is the trans-conductance of the i^{th}inverter and $i = a,b$. In particular, if the cross-coupled inverters are symmetric, $\tau_a = \tau_b = \tau$, $v_{ma} = v_{mb}$ and in terms of the difference voltage $v_D(t) = v_a(t) - v_b(t)$ we get $\tau \dot{v}_D(t) = v_D(t)$ the solution of which is

$$v_D(t) = v_D(0)e^{t/\tau} \tag{3}$$

From a transient simulation of the resolving nodes ($v_D(t)$), the exponential rate of divergence τ can be computed. The result of such a simulation using a symmetric latch circuit as the one shown in Fig. 1a (symmetric inverter and same size transistors with respect to nodes *a,b*) is shown in Fig. 2. At 1nsec the voltage controlled switch is opened and nodes V_a and V_b diverge to opposite directions (black solid lines). The logarithm of the voltage difference $v_D(t)$ is plotted in blue, clearly showing an exponential resolution in time as predicted by Eq. (3). The inverse of the derivative of the blue line is shown in green yielding τ. The flatness of the green line corresponds with Eq. (3).

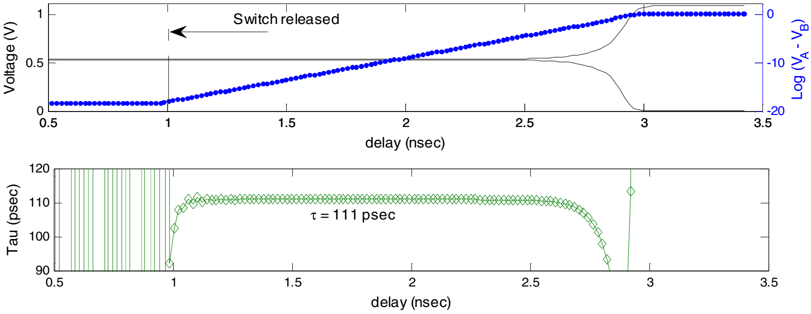


Fig. 2. Node shorting simulation for symmetric latch shown in Fig. 1a

Without the assumption of symmetry in the latch, the general solution of Eq. (2) is

$$\begin{aligned} v_a(t) - v_{mb} &= v_{1+}e^{t/\tau} + v_{1-}e^{-t/\tau} \\ v_b(t) - v_{ma} &= v_{2+}e^{t/\tau} + v_{2-}e^{-t/\tau} \end{aligned} \tag{4}$$

The constants $v_{1+}, v_{1-}, v_{2+}, v_{2-}$, are determined by initial conditions and depend on the setting of origin of the time scale and $\tau = \sqrt{\tau_a \tau_b}$. In this case, shorting the nodes does not force a metastable state in the latch, and hence this simple procedure cannot be used to simulate τ. Fig. 3 shows a simulation using the latch configuration of Fig. 1a with non-symmetric inverters, using the same color code as in Fig. 2. The green plot is not flat, showing non simple exponential behavior and hence τ cannot be computed from the slope of the logarithm of the voltage difference as proposed by the simple short node simulation method.

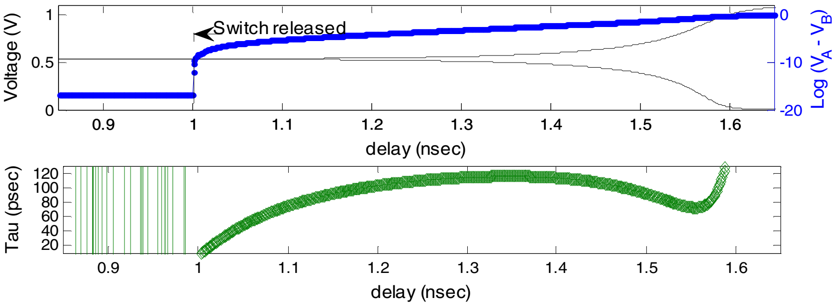


Fig. 3. Node shorting simulation of non-symmetric latch of Fig. 1a

When the cross-coupled inverters and capacitance loading are symmetric in the circuit of Fig. 1a, the metastable point lies on the line $V_a = V_b$ (Fig. 4a) and when the nodes are shorted the system is forced into metastability (blue circle). On the other hand, when the cross-coupled inverters are non-symmetric (Fig. 4b, skewed low, or Fig. 4c, skewed high), the metastable point is not reached by shorting the two nodes. Instead, shorting the two nodes yields an intermediate state (green circle), different than the metastable state (blue circle); when the switch is opened, the latch follows the green path in state space, from the blue circle on $V_a = V_b$ towards either the (1,0) state (Fig. 4b) or the (0,1) state (Fig. 4c).

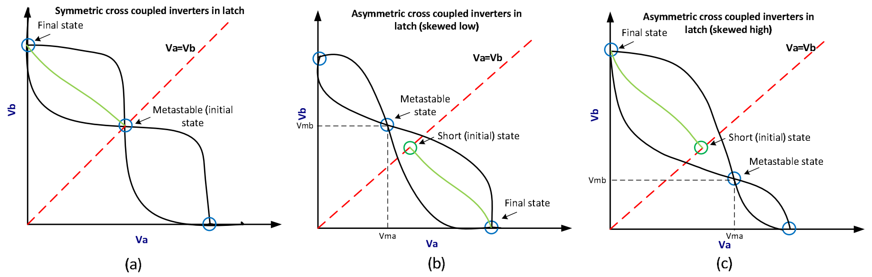

Symmetric cross coupled inverters in latch
Final state
Va=Vb
Metastable (initial state
Vb
Va
(a)
Asymmetric cross coupled inverters in latch (skewed low)
Va=Vb
Metastable state
Short (initial) state
Final state
Vb
Vmb
Vma
Va
(b)
Asymmetric cross coupled inverters in latch (skewed high)
Final state
Va=Vb
Short (initial) state
Metastable state
Vb
Vmb
Vma
Va
(c)

Fig. 4. Voltage transfer curves (VTC) (a) Symmetric latch (b)skewd low asymmetric latch (c) skewed high asymmetric latch

3 Extended Node Shorting Simulation

In an asymmetric latch, the metastable voltages of the two nodes differ by some V_{diff} (Fig. 5a), which needs to be found. Using the notation of (4), $V_{diff} = v_{ma} - v_{mb}$. If a voltage source $V_S = V_{diff}$ is placed between the metastable nodes, shown in Fig. 5b, when the switch is closed the latch is forced into metastability (blue circle). Then the switch can be released showing the exponential behavior predicted by Eq. (4). In the case when the value of V_S is exactly V_{diff} the current through the switch is zero, and thus the switch can be opened without changing any condition. This is caused because the intrinsic difference between the metastable voltages of the nodes (V_{diff}) is compensated for by the voltage source $V_S = V_{diff}$, resulting in no current through the switch.

Consequently our enhanced node shorting simulation (ENSS) method comprises two steps:

(i) Finding the metastability offset voltage (V_{diff}).
(ii) Node shorting transient simulation as described in Sec. 2 using $V_S = V_{diff}$.

An iterative process is used to find the value of V_{diff}. An adjustable voltage source V_s is used, and its value is changed until V_s=V_{diff}, namely until the metastable point lies on the line V_s=V_{diff}=$V_a - V_b$ (Fig. 5a).

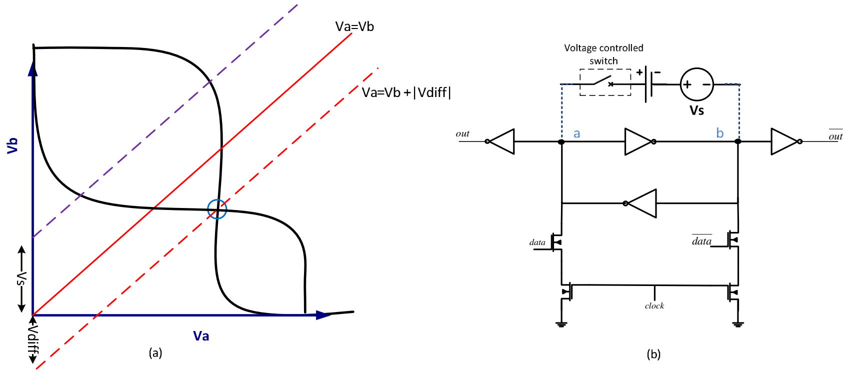


Fig. 5. Proposed technique for reaching metastability in asymmetric latches

We propose three different iterative algorithms to calculate V_{diff}:

- Current compensation (CC)
- Transient bisections (TB)

The current compensation algorithm adjusts the voltage V_s with the switch closed, using the circuit of Fig. 5b with an arbitrary initial value of V_s. If V_s>V_{diff}, current flows in one direction, and if V_s<V_{diff}, current flows in the opposite direction (Fig. 6). The algorithm iteratively adjusts V_S until the current is zero. At that stage,

$V_s = V_{diff}$ and the latch is metastable. A pseudo-code describing the algorithm is shown in Alg. 1. It starts with two initial voltages (V_{i+}, V_{i-}) yielding currents (I_s) with opposite sign. Then the algorithm performs a binary search untill the current falls below the desired error tolerance (ε). The value of V_{diff} is given by the last value of $(V_{i+} + V_{i-})/2$. The algorithm uses only SPICE DC simulations .

Algorithm 1. Calculate V_{diff} using CC

```
Require: $V_{i+}$ leads to $I_s \geq 0$
Require: $V_{i-}$ leads to $I_s < 0$
while $(|I_s| \geq \epsilon)$ do
    $V_s \leftarrow (V_{i+} + V_{i-})/2$
    $DC\_sim(V_s)$
    if $(I_s \geq 0)$ then
        $V_{i+} \leftarrow (V_{i+} + V_{i-})/2$
    else
        $V_{i-} \leftarrow (V_{i+} + V_{i-})/2$
    end if
end while
```

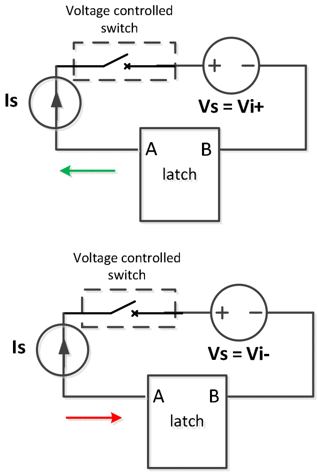


Fig. 6. Illustration of CC circuit diagram and behavior

In the transient bisection method, SPICE transient simulations are used with the circuit of Fig. 5b. The algorithm starts by choosing two values for V_s, $\{V_{i+}, V_{i-}\}$ which lead to two opposite transitions of the node V_a (or V_b). The transition direction is determined by the value of the voltage at the end time of the simulation (T). The transition settling time (T_S), measured from the time when the switch is released is computed, and the resolution is given by the maximum settling time allowed (MAX_{T_S}). Next, by binary search the algorithm finds a narrower interval, which also produces two opposite transitions on its extremes. A pseudo code for the algorithm is shown in Alg. 2

Algorithm 2. Calculate V_{diff} using TB

```
Require: $V_{i+}$ leads to $V_a(t = T) == V_{dd}$
Require: $V_{i-}$ leads to $V_a(t = T) == 0$
while $(T_s \leq MAX_{T_s})$ do
    $V_s \leftarrow (V_{i+} + V_{i-})/2$
    $transient\_sim(V_s)$
    if $(V_a(t = T) == V_{dd})$ then
        $V_{i+} \leftarrow (V_{i+} + V_{i-})/2$
    else
        $V_{i-} \leftarrow (V_{i+} + V_{i-})/2$
    end if
end while
```

Once the value of V_{diff} is found, with either one of these two methods, a single transient simulation is performed with the circuit of Fig. 5b. The voltage source V_s is set to the found value of V_{diff} and the switch is opened showing a divergence of the metastabale nodes. Re-writing Eq. (4) for the voltage node difference (v_D)

$$\begin{aligned}\underbrace{v_a(t) - v_b(t)}_{v_D(t)} + \underbrace{v_{ma} - v_{mb}}_{V_{diff}} &= (v_{1+} - v_{2+})e^{t/\tau} + (v_{1-} - v_{2-})e^{-t/\tau} \\ &\approx (v_{1+} - v_{2+})e^{t/\tau}\end{aligned} \tag{5}$$

The negative exponent term in Eq. (5) decreases fast with time and can be neglected. Fig. 7 shows a transient simulation of the circuit of Fig. 5 after finidng V_{diff} using the current compensation algorithm. As predicted in Eq. (5) the voltage ($v_D + V_{diff}$) shows exponential behavior and τ can be calculated from the slope of the blue curve (green line). When v_D is larger than about 100 mV the small signal model fails and the traces are no longer exponential as can be seen for times greater than 2.2nsec in Fig. 7.

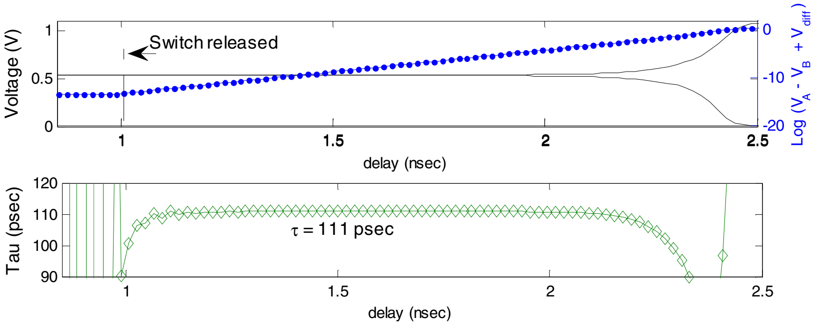


Fig. 7. Extended short node simulation for asymmetric latch using current compensation method to find V_{diff}

Though the two proposed methods generate consistent results for V_{diff} and hence for τ, the CC method incurs less computational effort and requires simpler simulations than the bisection method. CC method require DC simulations only, while TB requires several transient SPICE simulations with fine resolution and long run times. In order to achieve the same resolution for V_{diff} in the TB method, the transient simulation time in each run should be increased and hence increasing the overall simulation time. For that reason our tested ENSS uses CC to calculate V_{diff}. The ENSS using CC proposed is more time efficient than the sweep simulation method [8] since it requires fewer steps of much simpler DC simulations compared to several transient simulation, iterations and interpolations required in the sweep method.

3.1 Metastability Time Window T_W

The drawback of the ENSS method is its inability to simulate the parameter T_W, required in Eq. (1) to calculate failure probability. In most cases, however, knowing the value of τ is sufficient to reliably estimate the failure probability and a lower bound on MTBF can be found. Since T_W is the smallest data input arrival time window such that for all data toggling outside this window the settling time of the latch does not increase above its nominal value [9], then $T_W \leq t_{setup} + t_{hold} \leq T_C$. Then the error probability can be re-written as:

$$MTBF \geq \frac{e^{S/\tau}}{\left(t_{setup} + t_{hold}\right) \times F_C \times F_D} \geq \frac{e^{S/\tau}}{F_D} \tag{6}$$

Both last terms of Eq. (6) are good lower bounds for the design of reliable synchronizers using the simulation method proposed to obtain τ.

3.2 Multi Stage Synchronizers

The method derived so far applies only to a single latch. The calculation of the failure probability of a synchronizer comprising multiple cascaded latch stages from its constituent latch parts is given in [10]. A detailed study using our enhanced method is out of the scope of this work and will be addressed in future publications.

4 Simulations and Measurements

A library latch (Fig. 8) has been implemented in a 65nm LP CMOS process. Its performance has been measured and the results are compared here to our simulations based on the ENSS method, as well as to results generated by two other state of the art simulation methods, the sweep simulation method [8] [9] and the parametric simulation method [10]. CC method for calculating V_{diff} is used. The library latch is asymmetric due to different loading of the two latch nodes. All simulations were performed using SPICE BSIM4 model level 54. The measurement method was described in [11]

A comparison of τ in measurements and simulations is presented graphically in Fig. 9 for different levels of supply voltage between 0.95V and 1.3V, at room temperature. Fig. 10 shows the percentage error difference between each of the simulation methods and the measurements results of τ. Note that all three methods yield consistent values for τ with a maximum error of 11.5% with respect to measured values. The results of the proposed simulation method (ENSS) fall within 3% of the results of the other methods tested.

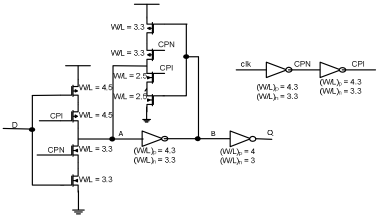


Fig. 8. Library latch used for simulation and measurements

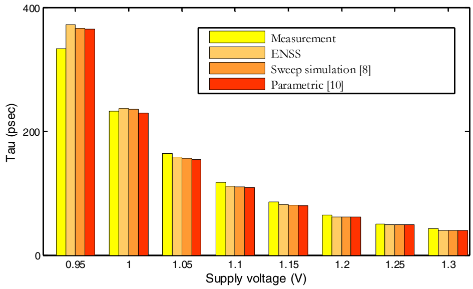


Fig. 9. $\boldsymbol{\tau}$ vs supply voltage, for measurements and simulations of a library 65nm CMOS FF

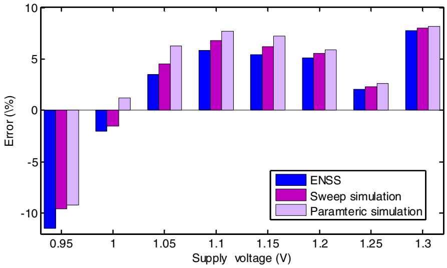


Fig. 10. Simulation errors with respect to measurements for different supply voltages

A comparison of the run times for ENSS simulation and the sweep simulation is shown in Fig. 11. For a fair comparison all simulations were performed using a common maximum resolution time (T_S). The V_{diff} resolution of ENSS was previously calibrated for T_S. This is why for higher supply voltages, for which τ is lower, more iterations are required to achieve the target resolution time and hence the run time is higher. The results show that our method provides accurate results much faster than the sweep and parametric method.

For the sake of completeness, Fig. 12 shows the offset of the metastable point from the symmetrical case, V_{diff}, against supply voltage. Note that V_{diff} is never zero along the range of supply voltage.

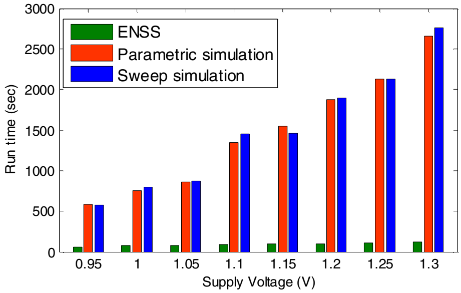


Fig. 11. Run times for ESNS simulation method and sweep simulation method

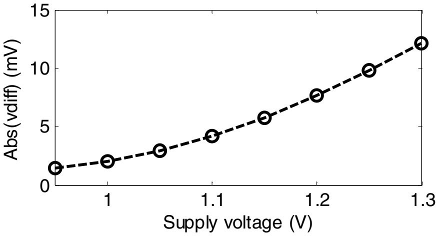


Fig. 12. Offset of metastable point ($\boldsymbol{V_{diff}}$) against supply voltage

5 Conclusions

We demonstrated that the node shorting simulation method as previously used in the literature is inappropriate for simulating latches and only works in the very special case of perfectly symmetric latches. We extended the node shorting simulation method for the case of non-symmetric latches and showed that it produced consistent results. The extended proposed method comprises two steps, finding the metastable offset voltage V_{diff} followed by a single run of a transient simulation. We showed two different algorithms to calculate V_{diff}, namely the current compensation, and the transient bisection method. We compared the results of our extended simulation method with the sweep and parametric simulation methods and showed that the results match with high accuracy but incurs less computation time and using only DC SPICE simulations followed by one transient simulation. We validated our simulation method against measurements taken on a circuit fabricated in a CMOS LP 65nm process. Simulation results predict τ with an error of less than 12% (measurement equipment error) compared to measurements, demonstrating that the proposed simulation method is suitable for characterizing synchronizers in a reliable and easy manner.

References

1. Kleeman, L., Cantoni, A.: Metastable behavior in Digital Systems. IEEE D&T 4(6), 4–19 (1987)
2. Dike, C., Burton, E.: Miller and noise effects in synchronizing flip-flop. JSSC 34(6), 849–855 (1999)
3. Beer, S., Ginosar, R., Priel, M., Dobkin, R., Kolodny, A.: The Devolution of Synchronizers. In: ASYNC 2010 (2010)
4. Chen, D., Singh, D., et al.: A comprehensive approach to modeling, characterizing and optimizing for metastability in FPGAs. In: FPGA 2010 (2010)
5. Zhou, J., Kinniment, D., Russell, G., Yakovlev, A.: Adapting synchronizers to the effects of on chip variability. In: ASYNC 2008 (2008)
6. Cox, J., Engel, G.L.: Metastability and Fatal System Errors. Blendics LLC (2010), `http://www.blendics.com`
7. Kinniment, D.: Synchronization and Arbitration in Digital Systems. Wiley (2007)
8. Yang, S., Greenstreet, M.: Computing synchronizer failure probabilities. In: DATE 2007 (2007)
9. Jones, I.W., Yang, S., Greenstreet, M.: Synchronizer Behavior and Analysis. In: ASYNC 2009 (2009)
10. Cox, J., Chaney, T., Zar, D.: Simulating the behavior of Synchronizers, white paper, `http://www.blendics.com`
11. Beer, S., et al.: An on-chip metastability measurement circuit to characterize synchronization behavior in 65nm. In: ISCAS (May 2011)
12. Kawaguchi, H., Sakurai, T.: A reduced clock-swing flip-flop (RCSFF) for 63% power reduction. IEEE J. Solid-State Circuits (May 1998)
13. Levacq, D., Yazid, M., et al.: Half VDD Clock-Swing Flip-Flop with Reduced Contention for up to 60% Power Saving in Clock Distribution. In: ESSCIRC (September 2007)

Phase Space Based NBTI Model

Reef Eilers[1], Malte Metzdorf[1], Sven Rosinger[1],
Domenik Helms[1], and Wolfgang Nebel[2]

[1] OFFIS e.V., Escherweg 2, 26121 Oldenburg, Germany
reef.eilers@offis.de
[2] University of Oldenburg, 26129 Oldenburg, Germany

Abstract. A phase space based NBTI model that relies upon the well-known reaction-diffusion model is introduced. Temporal shift in threshold voltage and a new parameter called "healability" are used to characterize the state of the NBTI effect. The NBTI degradation is then simulated as a trace of the interpolated characterization parameters. Thereby, healability is crucial for the success of the model. The phase space based model is well suitable in performance oriented use cases, since a small deterioration of the simulation results comes with a vastly improved simulation speed. The additional phase space dimensions of temperature and supply voltage in combination with the conversion between duty cycle and supply voltage permit a performance efficient way to simulate NBTI degradation for complete circuits without disregarding power gating, temperature profiles and the IR drop.

1 Introduction

Variation, leakage and aging are major problems in current transistor technologies. Thereby, the aging effect and its implication on device lifetime is at least understood. Most important aging effects are negative bias temperature instability (NBTI), hot carrier injection (HCI) and electromigration. Among those effects NBTI has certainly the most complex characteristic. Therefore, it is very demanding to model the NBTI effect and its impact on transistor aging. Furthermore, the influence of the NBTI effect on other relevant device parameters, such as dynamic power and leakage, isn't analyzed, yet.

The NBTI part of integrated circuit design is currently based on worst-case assumptions. Thereby, each PMOS is thought to be always conducting [1], the IR drop is disregarded and the temperature is set to a high constant value. This leads to a vast amount of over-design and generates additional cost. This cost occurs due to an increased circuit area by the use of of additional redundancies or due to a decreased performance through additional slack.

In order to calculate the influence of the NBTI effect on relevant device parameters and to simulate the NBTI degradation for complete circuits without disregarding power gating, temperature profiles and the IR drop, the performance of the NBTI model is crucial. Therefore, we propose a method to simulate the NBTI effect in a performant way that comprises the complex characteristic of the effect. Thus, some of the worst-case assumptions in integrated circuit

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 52–61, 2013.

design can be disregarded and the NBTI aging effect can be predicted in a less pessimistic way.

The paper is organized as follows. The basics of our NBTI simulation method are presented in Section 2. In Section 2.3 an example of a calculated phase space, which forms the base of our simulation method, is shown. Simulation examples and evaluation results follow in Section 3. Section 4 finally concludes the paper.

2 Phase Space Based NBTI Model

The main idea of the phase space based NBTI model is to characterize the state of the NBTI process of a transistor using only a few parameters. The transistor degradation for a given set of outer conditions can than be simulated by interpolating the alteration of these parameters based upon a set of precalculated alteration values. This set of alteration values is called phase space within this paper. All alteration values of the phase space are calculated using a specific time step. The transistor degradation is then simulated as a trace of interpolated characterization parameters (see Figure 5). Resolution of the trace is thereby dependent on the time step of the phase space.

The calculation of the phase space must be based upon an NBTI model. Therefore, the reaction-diffusion model of [2] [3] was chosen. The differential equations that define this model are presented in Equations 1-4. Since we had no access to reaction-diffusion parameters based upon a current technology, model parameters are taken from [2] and correspond to accelerated transistor aging. The reaction-diffusion model is based on hole assisted breaking of Si-H bonds in the Si-SiO_2 interface [4]. The released hydrogen converts to molecular hydrogen [3] and diffuses into the gate oxide. This model is well in line with measurements performed until 2005/2006. However, recent measurements [1] show that the short time effects and the long time recovery can't be explained by the reaction-diffusion model. Instead, this may give evidence to hole trapping being the cause of NBTI. Based on these results the switching trap model was proposed [5] [6]. This model may be the successor of the reaction-diffusion model. However, up to now there is still no consensus regarding the physical origin of NBTI [6]. However, the main ideas of the phase space based model should also be applicable to the switching trap model. Therefore, it should be possible to adapt the phase space based NBTI model to a newer base NBTI model.

Main advantage of the phase space approach is simulation speed. Furthermore, a change of the outer conditions at an advanced point in time doesn't require to start the simulation from the very beginning with a totally unaged transistor. Of course, this comes at the cost of calculating the whole phase space. However, this has to be done only once for each technology and may be processed in parallel. There might also be a problem regarding the interpolation accuracy, which will be addressed in the evaluation in Section 3. Another immanent problem of this simulation approach is the time step of the phase space. Changes of outer conditions at rate faster than the time step are obviously not supported and very short time steps may increase the interpolation problem.

The dimension of the phase space is given by the number of parameters for the characterization plus the number of parameters describing the outer conditions. Within this paper, temperature, supply voltage and transistor duty cycle are used for the outer conditions. Duty cycle can be directly used within an NBTI model during the calculation of the phase space and stands for a measure of activity. Two parameters are used for the characterization of the state of the NBTI process. The first parameter is the temporal component of the shift in threshold voltage. This is equivalent to the number of hydrogen molecules within the gate oxide in the case of the reaction-diffusion model. The second parameter should characterize the ability of the system to regenerate. It is therefore called "healability" and is linked to the probability that the temporal component of the shift in threshold voltage becomes permanent. The two parameters should also be independent of one another. These requirements led to a definition of the healability that is described in Section 2.1. The permanent component of the shift in threshold voltage is treated specially. During a simulation it can only increase over time. As long as the permanent shift in threshold voltage is small in comparison to the maximal possible shift in threshold voltage, the calculations of the reaction-diffusion model are barely affected by a permanent voltage shift. Therefore, it was decided to not use the permanent shift in threshold voltage as a third characterization parameter. Instead, the permanent voltage shift is calculated for every phase space entry. During a phase space simulation the permanent shift in threshold voltage that occurs in each time step is then summed up.

2.1 Healability

The healability definition is based upon a solution of the system of differential equations (Eq. 1-4) that characterize the reaction-diffusion model. These equations are presented in [2] and are slightly adapted within this paper to comprise the conversion to molecular hydrogen [3].

$$\frac{dN_{IT}}{dt} = k_F(N_0 - N_{IT}) - 2k_R N_H N_{IT} \qquad x = 0 \tag{1}$$

$$\frac{dN_{IT}}{dt} = 2D_H \frac{dN_H}{dx} + \delta \frac{dN_H}{dt} \qquad 0 < x < \delta \tag{2}$$

$$D_H \frac{d^2 N_H}{dx^2} = \frac{dN_H}{dt} \qquad \delta < x < T_{ox} \tag{3}$$

$$D_H \frac{dN_H}{dx} = -k_P N_H \qquad T_{ox} < x \tag{4}$$

where N_{IT} is the number of interface traps, N_0 is the initial number of unbroken Si-H bonds, N_H is the hydrogen concentration, k_F is the forward dissociation rate constant, k_R is the annealing rate constant, k_P is the surface recombination velocity, D_H is the hydrogen diffusion coefficient, $x = 0$ denotes the Si-SiO$_2$ interface, δ is the interface thickness and T_{ox} is the oxide thickness.

The following approach is used to solve the system of differential equations.

$$N_H(x,t) = \begin{cases} -m(t)x + b(t) + \kappa(x,t) \\ A(t)e^{-\beta(t)x} \end{cases} \tag{5}$$

$\kappa(x,t)$ stands for the deviation between linear solution and current system state. Inserting the approach in Equation 4 gives

$$\beta(t) = \frac{k_P}{D_H} \tag{6}$$

Continuity in T_{ox} yields

$$A(t) = \left(-m(t)T_{ox} + b(t)\right) e^{\frac{k_P}{D_H} T_{ox}} \tag{7}$$

Continuity of the derivative in T_{ox} yields

$$b(t) = m(t)\left(\frac{D_H}{k_P} + T_{ox}\right) \tag{8}$$

Integration of $N_{IT}(t) = \int_0^\infty 2N_H(x,t)dx$ delivers

$$m(t) = \frac{N_{IT}(t) - 2\int_0^\infty \kappa(x,t)dx}{\left(T_{ox}^2 + 2\frac{D_H T_{ox}}{k_P} + 2\frac{D_H^2}{k_P^2}\right)} \tag{9}$$

The linear solution $\kappa(x,t) = 0$ of the concentration of hydrogen molecules $N_H(x,t)$ is therefore thoroughly defined by the number of interface traps N_{IT}. In this way, there is a certain linear solution for every shift in threshold voltage. This solution is shown as a blue line in Figure 1 for an example value of the threshold voltage shift.

The calculation of the healability value for a given hydrogen concentration starts with the linear solution of the corresponding threshold voltage shift. Next step is a weighted difference between hydrogen concentration and linear solution. The weight function is a linear function with the value 1 at Si-SiO$_2$ interface ($x = 0$) and the value -1 at the end of the oxide ($x = T_{ox}$). Thus, in the first half of the oxide a hydrogen concentration that is higher than the linear solution produces positive values of the weighted difference in this region. Likewise, a hydrogen concentration that is lower than the linear solution produces positive values in the second half of the oxide. This is illustrated by the arrows in Figure 1. As shown in Equation 10, the healability is then defined as the spatial integral of the weighted difference divided by the spatial integral of the hydrogen concentration.

$$\text{Healability} = \frac{\int_0^{T_{ox}} (N_H - LS)\cdot WF\ dx}{\int_0^{T_{ox}} N_H\ dx} \tag{10}$$

where LS represents the linear solution of the system of differential equations (Eq. 5, 8 and 9) and WF is the weight function. In this way, healability is restricted to the region $[-1,1]$. Examples of hydrogen concentrations with positive and negative healability values are given in Figure 1.

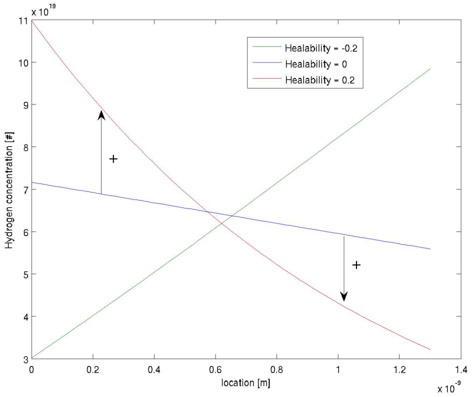


Fig. 1. Three examples of concentration of hydrogen molecules within the gate oxide. The examples have the same shift in threshold voltage (integral) but different healability values. The blue curve (Healability = 0) is the linear solution of the system of differential equations. Difference between hydrogen concentration and linear solution defines the healability value.

In order to calculate the phase space it is needed to generate a hydrogen concentration based upon a given shift in threshold voltage and healability. This is done starting with the linear solution of the shift in threshold voltage. Afterwards, the hydrogen concentration is changed repeatedly in minor steps in the first and second half of the oxide as long as the desired healability is reached. At each position, the change of the hydrogen concentration is a product of the appropriate hydrogen concentration value of the previous iteration and a predefined multiplier. Thereby, the spatial integral of the hydrogen concentration is kept constant.

2.2 Duty Cycle Conversion

Duty cycle can be directly used within the reaction-diffusion model and stands for a measure of activity. Hence, the effect of power gating or different signal probabilities on NBTI degradation can be simulated with the duty cycle parameter. The reaction-diffusion simulation results are independent of the duty frequency in wide frequency range [7]. Therefore, duty frequency is irrelevant as long as it is within this range.

The dimension of the phase space is given by the number of characterization parameters (temporal shift in threshold voltage and healability) plus the number of parameters describing the outer conditions. Within this paper, temperature, supply voltage and duty cycle should be used for the outer conditions. Therefore, the parameter duty cycle adds a phase space dimension and greatly increases the effort to calculate the phase space. However, effects of supply voltage and duty cycle may also be specified in a combined way with a single phase space dimension. This could be achieved with an interpolation that maps the plane

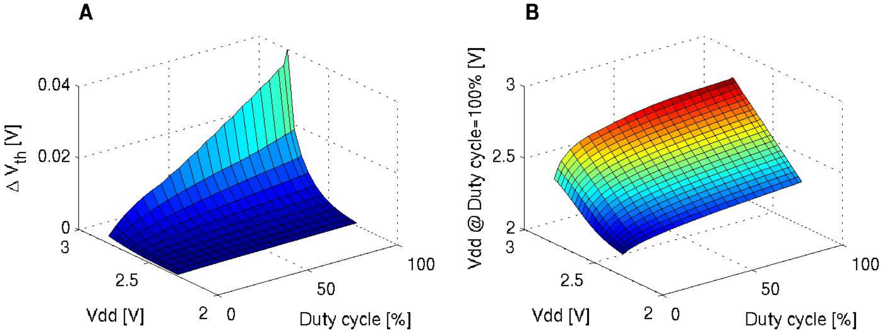


Fig. 2. Conversion between duty cycle and supply voltage. Panel A: Shift in threshold voltage simulated by the reaction-diffusion model. Panel B: Interpolated supply voltage at 100% duty cycle that yields the same shift in threshold voltage.

of supply voltage and duty cycle to a single dimension of corresponding supply voltage at a precise duty cycle value. This interpolation is called duty cycle conversion within this paper. It is based upon the reaction-diffusion results of the shift in threshold voltage as a function of supply voltage and duty cycle. These results are shown in panel A of Figure 2. In this way, the interpolation of a corresponding supply voltage at 100% duty cycle, that yields the same shift in threshold voltage as the different pairs of supply voltage and duty cycle, can be performed. The interpolated supply voltage values are presented in panel B of Figure 2. Within this paper, a single dimension of corresponding supply voltage at 100% duty cycle is used during the phase space calculation instead of separate dimensions of supply voltage and duty cycle.

2.3 Phase Space

A calculated phase space is presented for a time step of one minute and a corresponding supply voltage of 2.7 V. As stated in Section 2, these parameter values are due to reaction-diffusion model parameters only being available for accelerated aging of an outdated technology. Since the supply voltage only affects the general amount of the NBTI effect, it was chosen to restrict the number of supply voltage values. Results are presented in two different ways in Figure 3 and 4. The arrows in Figure 3 show the directions of the changes that occur in a minute within the plane of temporal shift of threshold voltage and healability. The actual differences between initial values and calculated values after a time step are presented in Figure 4. Thereby, the components of temporal shift in threshold voltage and healability are shown in panel A and B, respectively.

The directions of the arrows are dependent on both temporal shift in threshold voltage and healability. Since the arrows at the edge of the plane of threshold voltage and healability are mainly directed inwards, it is very unlikely that the precalculated plane is abandoned during a simulation. This is a major prerequisite for the reliability of the simulation technique. While the arrow directions in Figure 3 are rather independent of temperature, the length of the arrows and therefore the

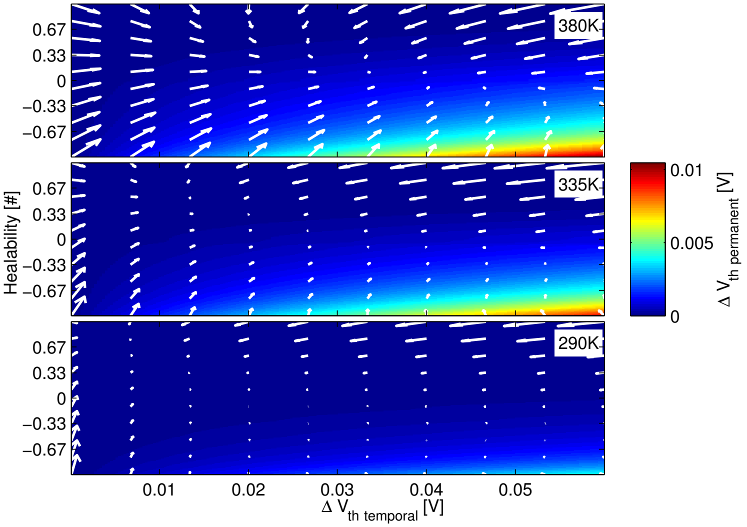


Fig. 3. Calculated phase space (white arrows) and permanent component of the shift in threshold voltage (colors) for a time step of one minute. Different panels represent temperatures of 290, 335 and 380 K. The arrows are scaled in order to fit within the plot. The correct differences of the threshold voltage and healability components that occurred within a time step are shown in 4A and 4B, respectively.

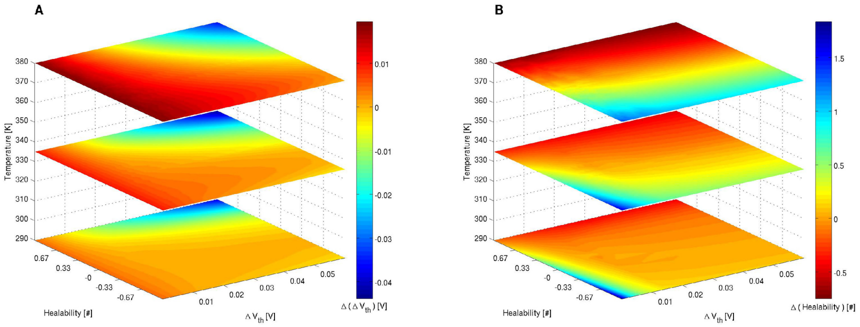


Fig. 4. Temporal shift in threshold voltage (Panel A) and healability component (Panel B) of the calculated phase space. The differences between initial values and calculated values after a time step of one minute are shown. Red colors represent increased transistor aging.

amounts of the changes are certainly not. This equates to the temperature activation of the NBTI effect. The difference in threshold voltage is highly dependent on temporal shift in threshold voltage and healability. This is best visible at the highest temperature in Figure 4A. In this case, a reduction of the voltage shift only occurs at higher values of the initial voltage shift and positive healability values. In contrast, the difference in healability at the highest temperature in Figure 4B is only dependent on the initial healability value. At lower temperatures a dependency on temporal shift in threshold voltage becomes apparent. However, this dependency is only present at lower voltage values. This result shows that the parameter healability is independent of the parameter temporal shift in threshold voltage to a certain degree. This independence was a goal of the healability definition.

The permanent components of the shift in threshold voltage that occur during the time step of one minute are shown as colormap in Figure 3. These components are summed up during a phase space based simulation. Relevant values of the permanent component only occur at negative healability values. This result also confirms the healability definition in Section 2.1. As expected, the permanent shift in threshold voltage also increases with an increasing value of the initial temporal voltage shift.

3 Simulation Results

Parameter degradations of a transistor during one hour of permanent stress are simulated using the phase space based model. Thereby, the phase space of Section 2.3 with a time step of one minute is used. Hence, the simulations are composed of 60 interpolations within the phase space. A simulation example with a constant temperature of 290 K and an example with alternating temperatures are shown in Figure 5.

In order to evaluate the phase space based method, direct reaction-diffusion simulations are also performed for the conditions with one hour of permanent stress. The green arrow in Figure 5 shows the result of the direct reaction-diffusion simulation that is equivalent to the simulation example with a constant temperature of 290 K. In this example, there is only a marginal difference in shift of threshold voltage between these two methods. However, the simulation with the phase space based model was more than 600 times faster than the

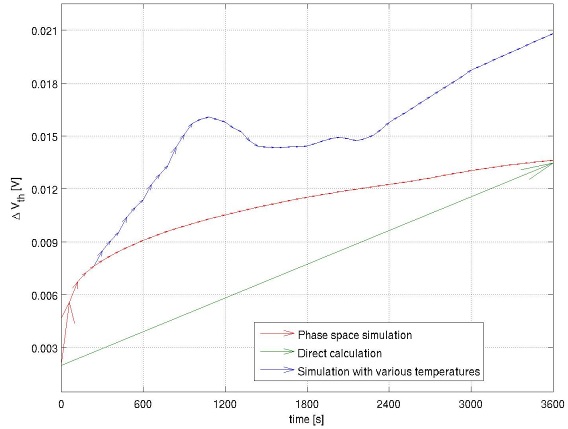


Fig. 5. Simulation of one hour of permanent stress. Red arrows represent the phase space simulation, while the green arrow represents a direct calculation with the reaction-diffusion model at the same temperature. An example of a phase space simulation with different temperatures is shown by the blue arrows.

direct reaction-diffusion simulation. In this way, simulations that would need about a month with the reaction-diffusion model can be calculated in less than 75 minutes. The short time step of the phase space was chosen, since the parameters of the reaction-diffusion model correspond to accelerated transistor aging. If we had access to reaction-diffusion parameters based upon a current technology, a phase space with a greater time step could have been used. In this case, the benefit in simulation speed would have been greater.

Direct reaction-diffusion simulations are performed for a wide range of initial shift in threshold voltage and temperature. Percental differences between the direct simulations and the phase space simulations are calculated for all those conditions. These results are presented in figure 6. The different panels of the figure illustrate the benefit of the different features of the phase space based model. In panel A the phase space simulation is performed without using the healability parameter and the summation of the permanent component. A difference up to 40% occurs in this situation. This difference is reduced when the healability parameter (panel B) or the summation of the permanent component of the shift in threshold voltage (panel C) is used during the simulation. As expected, best phase space simulations are achieved when both healability parameter and

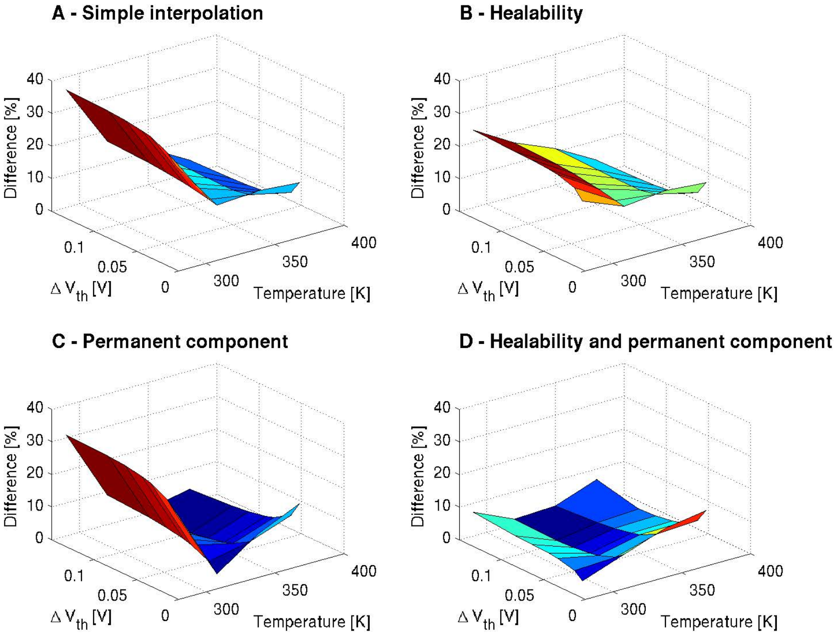


Fig. 6. Percental differences between direct reaction-diffusion calculation and phase space simulation of one hour of permanent stress. Panel A shows the differences for a simulation without healability parameter and summation of permanent component of the shift in threshold voltage. Simulation methods used in panel B and D make use of the healability parameter while the summation of the permanent component of the shift in threshold voltage is used in panel C and D.

summation of the permanent component are used (panel D). In this case, the percental difference between direct calculation and phase space simulation is always smaller than 10%.

4 Conclusion

The calculated phase space indicates that the new parameter "healability" is well defined with respect to the requirements. Furthermore, healability and summation of the permanent shift in threshold voltage are crucial for the success of the phase space based model. The model is well suitable in performance oriented use cases, since a small deterioration of the simulation results comes with a vastly improved simulation speed. The additional phase space dimensions of temperature and supply voltage in combination with the conversion between duty cycle and supply voltage permit a performance efficient way to simulate the dependence on various outer conditions. In this way, NBTI degradation can be calculated efficiently for complete circuits without disregarding power gating, temperature profiles and the IR drop.

Acknowledgments. This work has been funded by the RELY Catrene project (CA403).

References

1. Reisinger, H., et al.: A Comparison of Very Fast to Very Slow Components in Degradation and Recovery Due to NBTI and Bulk Hole Trapping to Existing Physical Models. IEEE Transactions on Device and Materials Reliability 7, 119–129 (2007)
2. Alam, M.A., Mahapatra, S.: A comprehensive model of PMOS NBTI degradation. Microelectronics Reliability 45, 71–81 (2005)
3. Alam, M.A., et al.: A comprehensive model for PMOS NBTI degradation: Recent progress. Microelectronics Reliability 47, 853–862 (2007)
4. Ushio, J., et al.: Electric-field dependence of negative-bias temperature instability. Journal of Applied Physics 97, 086101 (2005)
5. Grasser, T., et al.: Time-dependent defect spectroscopy for characterization of border traps in metal-oxide-semiconductor transistors. Physical Review B 82, 245318 (2010)
6. Reisinger, H., et al.: The statistical analysis of individual defects constituting NBTI and its implications for modeling DC- and AC-stress. In: International Reliability Physics Symposium, pp. 7–15 (2010)
7. Shen, C., et al.: Characterization and Physical Origin of Fast Vth Transient in NBTI of pMOSFETs with SiON Dielectric. IEDM Technical Digest 333 (2006)

Fast Propagation of Hamming and Signal Distances for Register-Transfer Level Datapaths

Axel Reimer[1], Lars Kosmann[1], Daniel Lorenz[1], and Wolfgang Nebel[2]

[1] OFFIS Research Institute
[2] University of Oldenburg,
D - 26121 Oldenburg, Germany
{reimer,kosmann,lorenz,nebel}@offis.de

Abstract. Hamming and signal distance are common model variables for characterising register-transfer level components for power. In order to use power estimation models based on this characterisation it is necessary to know exact input-stimuli for a circuit in order to propagate it through the datapath and calculate the Hamming and signal distances at each node.

In this work a new propagation approach for Hamming and signal distances is presented. It is based on precharacterised components for Hamming and signal distances. Using this approach it is not necessary to know the exact input data of a circuit, only statistical information of the input-stream is needed. The errors of the estimated Hamming and signal distance properties are in the range of 5 % to 14 % The result of an estimation is available nearly instantaneously since look-up tables are used for implementation.

1 Introduction

Power dissipation is a key parameter when designing integrated circuits since it influences costs, battery life and reliability of embedded systems. For this reason, a lot of research activity is located in the field of power estimation and power optimisation of digital circuits at different levels of abstraction.

Power dissipation can be separated into dynamic power, static power, and power dissipation caused by short-circuit currents. The sources of dynamic power are switched capacitances while leakage currents are the source for static power. In this paper we focus on dynamic power, more precisely on the estimation of switching activity.

The dynamic power is given by (1)

$$P_{dyn} = \frac{1}{2} \cdot \alpha \cdot C \cdot V^2 \cdot f \tag{1}$$

where α represents the switching probability, C is the capacitance to charge, V is the supply voltage, and f represents the frequency. The common method

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 62–71, 2013.

to estimate power on gate level or lower is to simulate a circuit by using a testbench and capturing the activity of the signals in a file. This file can be used with commercial power estimation tools and technology libraries of the vendor to get an estimation of the average power dissipation.

There are many approaches dealing with the estimation of power on register-transfer or behavioural level. One possible basis for an estimation on these high abstraction levels is to characterise the commonly used datapath components (adders, multipliers, etc.) for power to build macro-models for the components. In [1] the Hamming and signal distance between consecutive patterns are used to characterise typical ASIC datapath components for power. In [2] the same approach was used for FPGAs. The authors of [3] use the same model variables to estimate the power of an H.264 decoder. In general, a power estimation on the basis of Hamming and signal distance can be done but the estimator still needs input stimuli, propagate it through the datapath and calculate the Hamming and signal distance at each input of the used components. This means that the input stimuli are one important factor to get trustful results. But in many cases real stimuli are not known at this early stage.

There are some other papers dealing with the propagation of signal statistics through datapaths. The authors of [4] showed, that switching activity in datapaths can be characterised and propagated as probability based entropy of the used components. Further investigations on entropy based models were done in [5] by using a dual bit type model and switching activity statistics. Another approach in is done in [6], where statistical methods were used at word level to estimate the average switching activity of signals. To the best of our knowledge there exists no approach of propagating Hamming and signal distances through datapaths without knowing the datastream at the primary inputs.

In the following sections we present a method for propagating these signal characteristics. Power-models which are based on characterising components based on Hamming and signal distances can use this approach to abstract input stimuli to input-characteristics. Only a knowledge of the switching behaviour at the inputs is needed.

The paper is organised as follows. Section 2 explains the preliminaries needed for understanding the approach which is presented in the subsequent Sect. 3. In Sect. 4 the accuracy of the estimation is evaluated. The paper is concluded in Sect. 5.

2 Preliminaries

A transition between two consecutive bitvectors can be classified by the Hamming and signal distance. The Hamming distance Hd is defined as the number of bit-transitions between two consecutive bitvectors v_t and v_{t+1}:

$$Hd(v_t, v_{t+1}) = \sum_{i=1}^{bw} (v_t(i) \oplus v_{t+1}(i)) \tag{2}$$

where $\oplus$ is the XOR-Operator, bw is the bitwidth of the input vector and i is the bitposition in the vector.

The signal distance Sd is defined as the number of bits which are stuck at logical one in two consecutive vectors:

$$Sd(v_t, v_{t+1}) = \sum_{i=1}^{bw} (v_t(i) \wedge v_{t+1}(i)) \tag{3}$$

where $\wedge$ is a logical AND.

The maximum Hamming and signal distances of a bitvector transition are limited by the bitwidth of the vector:

$$Hd + Sd \leq bw \tag{4}$$

For a vector with a bitwidth of bw there exist 2^{bw} possible bitvectors. For every bitvector there are 2^{bw} possible consecutive bitvectors. This leads to (5) which defines the cardinality of the set of all possible transitions T for a vector with bitwidth bw.

$$card(T) = 4^{bw} \tag{5}$$

These bitvector-transitions can be separated into Hamming and signal distance classes. The set C of these classes has a cardinality of

$$card(C) = \sum_{i=1}^{bw+1} i \tag{6}$$

A propagation of an Hd/Sd-combination through a component can be done by feeding the component with every possible transition that represents this combination. The result is an output-distribution of Hamming and signal distances which can be represented in a table. For example, if we have a look at a 2-bit incrementer and an input Hd/Sd-combination of $Hd = 1$ and $Sd = 0$ the input and output transitions shown in Table 1 are possible:

Table 1. In- and output-transitions for a 2-bit incrementer with $Hd = 1$ and $Sd = 0$

input-transition ($v_t \rightarrow v_{t+1}$)	output transition ($v_t \rightarrow v_{t+1}$)	output Hd	output Sd
00 → 01	01 → 10	2	0
00 → 10	01 → 11	1	1
01 → 00	10 → 01	2	0
10 → 00	11 → 01	1	1

The according output-distribution for an Hd/Sd-input-combination can be represented by a table. Table 2 shows the distribution for the presented incrementer example. The fields which are invalid due to (4) are marked with a dash. Hereinafter, this Hd/Sd-output-distribution is called *output-triangle* since the valid fields form a triangular-shape.

In the following section the approach for propagating Hamming and signal distance classes through a datapath is presented.

Table 2. Hd/Sd-output distribution for a 2-bit incrementer with $Hd = 1$ and $Sd = 0$

hd\sd	0	1	2
0	0	0	0
1	0	2	-
2	2	-	-

3 Hd/Sd-Propagation Method

The Hd/Sd-propagation approach which is presented in this section uses the Hamming and signal distance at the inputs of an RT-component to estimate the output-triangle (see Sect. 2). The output-triangle then serves as input for propagating the Hamming and signal distance through the subsequent components of a datapath.

3.1 Validation of Data Abstraction

In order to validate whether the Hamming and signal distance are suitable parameters to be propagated through register-transfer level components, typical components (adder, multiplier) with a maximum bitwidth of eight per input were chosen since the number of possible input transitions rises by the factor of 16 for every additional bit at each input according to (5). The components where analysed by generating all possible input transitions for every Hd/Sd-combination and calculating the result. Afterwards the output-triangle for every possible input Hd/Sd-combination was generated.

If the Hamming and signal distance are suitable parameters then most of the distribution in the output-triangles should be accumulated in a few fields.

Figure 1 shows the resulting coverage (y-axis) of all input data by choosing a growing number of fields (x-axis) of the descending ordered list of the Hd/Sd-output mean distribution. Most of the output data is accumulated on a low

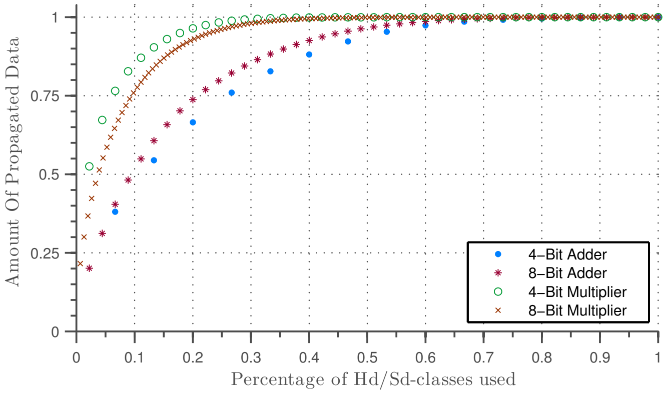


Fig. 1. Data coverage by number of fields of the output-triangle

percentage of the fields and with every additional field the coverage is rapidly increasing. Choosing a satisfying level of coverage at 75 %, this level is reached by using 22-27 % of fields of the adder and 7-10 % of the fields of the multiplier. This shows that a propagation of Hamming and signal distances is meaningful.

3.2 Monte Carlo Method for Hd/Sd-Propagation

The main objective of this method is to find the fields with high datacoverage for given input Hamming and signal distances. Since the quantity of input transitions increases rapidly by the bitwidth, exhaustive calculations would require a great deal of time and effort especially for common bitwidths e.g. 32 bit. Thus an approximation is needed. A commonly used method for exploring large amounts of data is the Monte Carlo method. It uses random data refined by a probability distribution to explore the experimental space. The accuracy can be controlled by the amount of random data, thus the error is a matter of paid effort. This approach uses the Monte Carlo method to find the fields with the highest probability for further propagation.

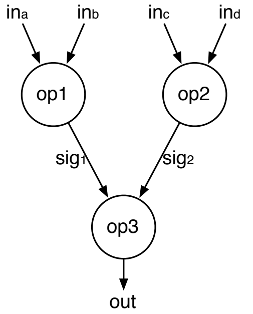


Fig. 2. An example datapath

The example datapath used is shown in Fig. 2. The Hamming and signal distances of the primary inputs serve as inputs for the method and can be given as one fixed value or as an distribution represented as matrix. The method iterates over given Hd/Sd-combinations at the input. For each iteration random data with the given Hamming and signal distance is generated and propagated through the first level by applying them to in_a to in_d and calculating the output triangle of sig_1 and sig_2. Then the fields with the highest probabilities are chosen and a new set of random data is generated for every possible combination of those Hd/Sd-classes. For every combination an output-triangle is calculated. Those output-triangles are then averaged using the probability of the occurence of the according input Hd/Sds of sig_1 and sig_2. The runtime depends on the quantity of random data generated for each Hd/Sd-combination and the number of fields which are chosen for the next propagation step.

For the implementation the quantity of random data is covered by a variable called *effort*. The *effort* represents the number of random input transitions per output field. For example, an 8-bit adder (two 8-bit inputs, one 8-bit output) has an output cardinality of $card(output) = 45$. Using an *effort* of 50 there are $45 \cdot 50 = 2250$ random input transitions generated for each characterised input Hd/Sd-combination. Each 8-bit input of the adder has a cardinality of $card(input_i) = 45$ which means that there are $45 \cdot 45 = 2025$ different Hd/Sd-combinations at the inputs. Since the adder has $2 \cdot 8 = 16$ input bits which can switch there are 2^{32} different transitions. This means that on average every Hd/Sd-class covers more than two million transitions. Thus only about 0.11 % of all transitions are generated for an *effort* of fifty in this example.

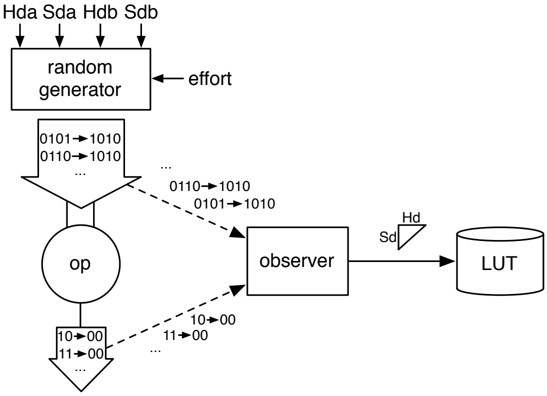


Fig. 3. Precharacterisation

3.3 Precharacterisation of Single Components

The calculation of the propagation approach can be done at runtime. But as particular components at steady bitwidths do not change their input related behaviour at the output for the same input data, the estimation time can be optimised by using precharacterisation as shown in Fig. 3. This can be done by a Monte Carlo exploration of a single components behaviour. The resulting output triangles are held as lookup-tables (LUTs) in a component database. By using this approach, the effort during estimation is reduced to the lookup process, as all behaviour data has been aggregated during the characterisation process.

The workflow of this approach is shown in Fig. 4. At the first level the primary Hd/Sd-combination is choosen by random or by predefined stimuli information. The output behaviour of sig_1 and sig_2 is looked-up separately as the behaviour of the single components are held in the LUT. The second level uses these output-triangles to look up the output-triangle of *out*. Since the result of the lookup-table is available nearly instantaneously the amount of propagated fields and thus the amount of probable data can be increased without decreasing the performance noticeable. Also the characterisation process can be done in advance with a large set of random input data, increasing the accuracy of the LUT-values.

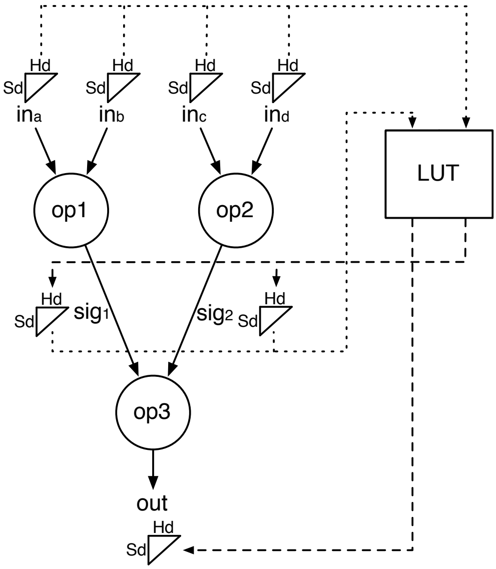


Fig. 4. LUT based approach

A restriction of the approach is that structural correlations are not taken into account. It is possible to estimate the Hamming and signal distances in circuits containing structural correlations but they are not specially treated.

4 Experimental Results

For the evaluation of both approaches, the estimation results have to compete against an exhaustive simulation since comparing a Monte Carlo method with another random data generation would not be sufficient. This limits the comparison to low bitwidths.

Measuring the error between estimated data and the evaluation model is done using the root mean square error (RMS). For a better comprehension the RMS is normalized by the range of observed values, that may be read as percentage value of error. Hereinafter it will be called NRMS by using the function shown in (7), where $\hat{x}$ represents the estimated value of the approach and x the result of an exhaustive simulation.

$$NRMS = \frac{\sqrt{\frac{\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}{n}}}{x_{max} - x_{min}} \tag{7}$$

Every Hd/Sd-combination at the input results in a matrix like shown in Table 2. Thus the estimation of a component or datapath includes a set of matrices. For a measure of the overall error, the NRMS between the exhaustive calculation and the Monte Carlo method is calculated for each input combination and averaged over all matrices, hereinafter called Mean NRMS.

One important parameter is the amount of necessary Monte Carlo data to get reliable results since it influences the runtime of the method. The influence of the *effort* to the accuracy of the estimation in shown in Fig. 5. It can be seen that an increasing *effort* directly results in a lower Mean NRMS. It also indicates that even with a small amount of random data, the Mean NRMS is acceptable low. For all evaluated operations an *effort* of 20 was enough to get a Mean NRMS as low as 5 % and less.

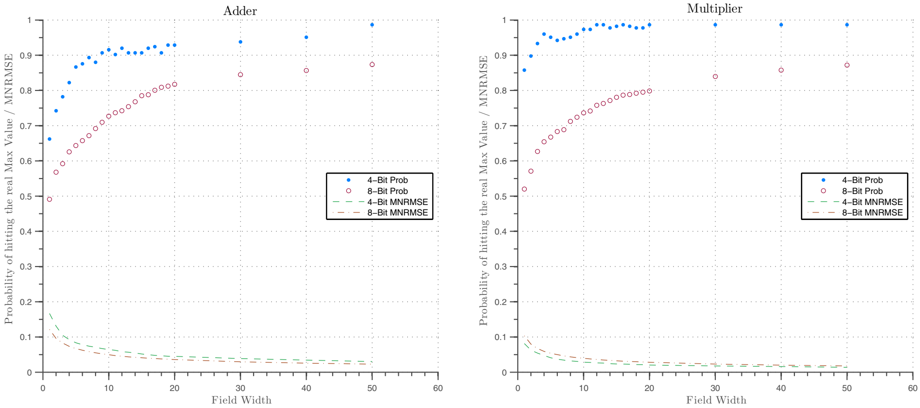


Fig. 5. Mean NRMS of used *efforts* and the probability of hitting a real maximum value

In order to use this approach for datapaths it is important to find the Hd/Sd-classes with the highest probability in the output-triangle. As can be seen in Fig. 5, the probability to find the most probable Hd/Sd-class is very high even with few random transitions. These results show that a Monte Carlo based characterisation is suitable for an approximation of the output-triangle.

The second important factor for datapath usage is the number of Hd/Sd-combinations used for propagation. To evaluate the impact of this value, the estimation of an example datapath as shown in Fig. 2 with an *effort* of 50 was done and compared against an exhaustive simulation. Figure 6 illustrates the error at *out* as Mean NRMS (y-axis) at different numbers of Hd/Sd-combinations propagated. In the datapath named *AdderDP* (*MultiplierDP*) the operations op_i are replaced by adders (multipliers). The datapath referred as *CombinationDP* realises the function $out = (a + b) \cdot (c + d)$. The number of fields propagated has direct impact in terms of reducing the mean NRMS for *AdderDP* and *CombinationDP*, while *MultiplierDP* does not benefit, anyway its mean NRMS is stuck at 14 %. If all fields are propagated there is still a error remaining. This is due to the used process of generating random data.

This number of propagated fields directly impacts the performance of estimation or, when precharacterisation is used, the time of characterising the components. The more Hd/Sd-combinations are propagated, the more random data

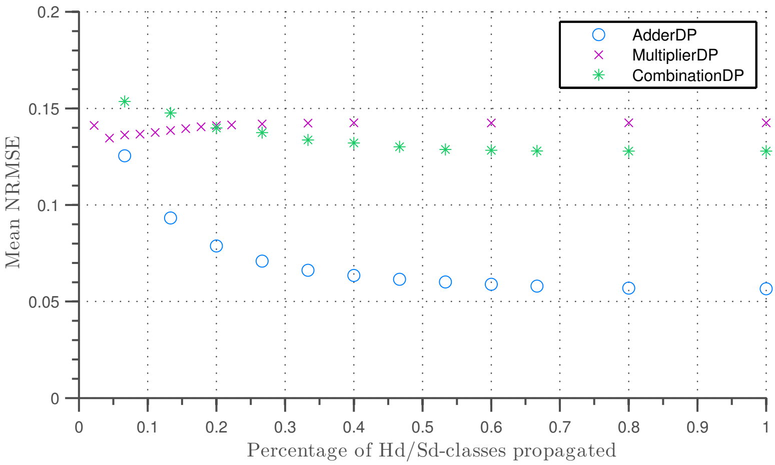


Fig. 6. Error of estimation with a different number of Hd/Sd-combinations

has to be generated and propagated. Moreover the Hd/Sd-combinations for the seperated inputs have to be combinated with each other.

The precharacterisation approach presented in Sect. 3.3 replaces the generation of the output-triangles at runtime by the lookup process. But as the lookup values are generated by the same method, the error is equivalent to the one presented in Fig. 6. The advantage is that more Hd/Sd-combinations can be propagated without a big performance penalty since this implies only more lookups without any need for generating new data. Another advantage in terms of increasing accuracy can be gained by using a high *effort* within the characterisation process in advance.

Table 3 shows the runtime of the precharacterisation process in comparison to an exhaustive calculation at two different *efforts*. The experiments were done on one core of a 2.4 GHz AMD Opteron processor. As can be seen for a bitwidth of four, the exhaustive simulation is more efficient. This is due to the fact, that the amount of input transistion randomly generated at an *effort* of 50 is 2.6 times higher than the amount of different transistions possible. This is a corner case. It would be more efficient to choose a much lower *effort* for this bitwidth. But at growing bitwidths this used *effort* results in a good fit of the characterisation

Table 3. Characterisation time for different bitwidth and *efforts* (*extrapolated)

	Adder			Multiplier		
Effort	50	500	exh.	50	500	exh.
4-Bit	1.0 s	3.9 s	0.7 s	1.7 s	10.9 s	0.8 s
6-Bit	5.7 s	36.3 s	88.1 s	13.9 s	117.8 s	79.0 s
8-Bit	25.0 s	189.5 s	8.7 h	67.5 s	10.8 h	7.9 h
16-Bit	0.4 h	4.1 h	*$16 \cdot 10^6$ h	1.5 h	15.1 h	*$15 \cdot 10^6$ y
32-Bit	37.7 h	16.1 d	*$4.8 \cdot 10^{27}$ h	6.3 d	59.1 d	*$4.4 \cdot 10^{27}$ y

data, as shown before. For 16- and 32-bit components, the exhaustive simulation is not practical. The runtime of the characterisation process is still acceptable since it has to be done only once for every component.

5 Conclusion

A Monte Carlo based approach for estimating the Hamming and signal distances throughout datapaths was presented. The generation of the random data can be done at runtime or in a precharacterisation process. When using precharacterisation the result of the estimation is available nearly instantaneously since it is based on lookup-tables. The estimation error for a reasonable efforts is 5 % or less for simple components and 14 % or less for two-level datapathes.

In the future we will focus on handling structural correlations and combining the approach with a Hamming and signal distance based power model. Additional improvements will be the extension by a glitch model and a faster characterisation by parallelizing the monte carlo process.

Acknowledgements. This work has been partially supported by the EnerSave Project, funded by the German BMBF under Grant Agreement 01BE1102.

References

[1] Jochens, G., Kruse, L., Schmidt, E., Stammermann, A., Nebel, W.: Power Macro-Modelling for Firm-Macros. In: Integrated Circuit Design, pp. 24–35 (2000)

[2] Reimer, A., Schulz, A., Nebel, W.: Modelling Macromodules for High-Level Dynamic Power Estimation of FPGA-based Digital Designs. In: Proceedings of the 2006 International Symposium on Low Power Electronics and Design, pp. 151–154 (2006)

[3] Park, Y.H., Pasricha, S., Kurdahi, F.J., Dutt, N.: System Level Power Estimation Methodology with H.264 Decoder Prediction IP Case Study. In: ICCD, pp. 601–608 (2007)

[4] Marculescu, D., Marculescu, R., Pedram, M.: Information Theoretic Measures for Power Analysis. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 599–610 (1996)

[5] Kyriakis-Bitzaros, E.D., Nikolaidis, S.: Estimation of Bit-Level Transition Activity in Data-Paths Based on Word-Level Statistics and Conditional Entropy. In: IEEE Proceedings of Circuits, Devices and Systems, pp. 234–240 (2002)

[6] Ramprasad, S., Shanbha, N.R., Hajj, I.N.: Analytical Estimation of Signal Transition Activity from Word-Level Statistics. IEEE Transations on Computer-Aided Design of Integrated Circuits and Systems, 718–733 (1997)

Noise Margin Based Library Optimization Considering Variability in Sub-threshold

Tobias Gemmeke[1], Maryam Ashouei[1], and Tobias G. Noll[2]

[1] Holst Center / imec, Eindhoven, The Netherlands
[2] EECS, RWTH Aachen University, Germany
tobias.gemmeke@imec-nl.nl

Abstract. Operating at near-threshold voltage is an attractive solution for energy reduction in wireless sensor nodes, where throughput is often in the order of MBaud. The challenges at low supply voltages are timing or functional failures due to variability. A common mitigation technique is to apply cell pruning based on stack height or complexity. We present a new variability-aware methodology based on the static unity gain noise margin and its corresponding closed-form expression that enables cell selection and/or optimization for reliable low-voltage operation. Using a standard cell library in a 40nm low-power CMOS process as vehicle, the minimal supply voltage could be lowered by 70mV, or the library complexity increased without impacting reliability.

Keywords: Standard cells, static noise margin, sub-threshold, random fluctuations, reliability, variability.

1 Introduction

Over the years, the demand for low power operations triggered by the widespread usage of portable devices drove down the supply voltage. This is because V_{DD} scaling is the most effective technique for reducing power due to the quadratic dependency of dynamic power on supply voltage [1]. In recent years, the rigid power consumption requirement of wireless sensor nodes (WSN) along with relatively low performance demands of their applications, cultivated a new wave of works aggressively scaling the supply voltage into sub-threshold regime [2]-[6].

While effective in reducing power consumption (e.g. a reduction of 4.7X was reported in [5]), operating in sub-threshold regime brings challenges in terms of severe frequency degradation as well as increased susceptibility to process variations. While the former is less of a concern for WSN with low performance requirements, the latter is an issue that must be addressed. This is to first guarantee functional correctness and second to avoid large design margin, which would reduce the benefit of sub-threshold operation.

Most of Systems-on-Chip designed in sub-threshold follow the general guideline of avoiding cells with high transistor stacks and complex gates (e.g. [3,4,6]). In this way functional failure due to low drive strength in the stack path and the larger drive strength in the complementary transistor network of a CMOS

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 72–82, 2013.

gate is avoided. Additionally, circuit-level simulations of individual cells help to identify problematic cells [7]. [8] and [9] propose new cell libraries taking into account logical effort and reverse short channel effect respectively. Authors in [10] consider process variability during cell design while keeping delay or area constant. The authors of [11,12] look at different logic styles.

Alioto [13] finds a closed form representation of static unity gain noise margin (NM) at sub-threshold and utilizes it to find the minimum operating voltage. In [14] a closed form equation is presented that defines the minimum operating voltage considering variability.

In this work, we present a novel methodology based on the unity gain noise margin that takes process variability into account to find the failure probability of each individual cell. This model is extended to predict the reliability of a complete design. The cell failure probability is used to guide the cell design to have a better yield for a given supply voltage or having a larger set of cells without impacting the yield or the functional supply voltage. It is further shown that avoiding the high transistor stacks does not necessarily yield the best cell library selection. The derived methodology enables a yield oriented library selection. Throughout the paper we use extensive Monte-Carlo (MC) circuit simulations relying on foundry provided statistical transistor models suitable for high-speed analog design. Sequential elements are beyond the scope of this paper, as their internal feedback loop requires stability metrics other than the one used.

The outline of this paper is as follows. First, we discuss how various design parameters impact the variability of the current in the sub-threshold regime. In Section 3, we provide an analytical model to estimate the failure probability. The application of the model to a standard cell library selection process is highlighted in Section 4. Section 5 concludes this paper.

2 Representing Variability in Sub-threshold

The driving current in the sub-threshold (sub-V_{th}) regime is I_{sub}, defined as [16]

$$I_{\text{sub}} = \beta(n-1) \cdot V_{\text{T}}^2 \cdot e^{\frac{V_{\text{GS}}-V_{\text{th}}+\eta V_{\text{DS}}}{nV_{\text{T}}}} \cdot (1 - e^{-\frac{V_{\text{DS}}}{V_{\text{T}}}}) \tag{1}$$

with sub-V_{th}slope $n, \beta = \dfrac{W}{L}\mu C_{\text{ox}}, V_{\text{T}} = \dfrac{kT}{q}$, drain-induced barrier lowering η.

Given the exponential dependency of I_{sub} on V_{th} and the normally distributed random variations of V_{th}, the I_{sub} distribution is log-normal. Hence, the discussion on variability is different from the super-threshold domain in which the parameter of interest such as delay has a normal distribution. In the next subsection, the confidence interval for a log-normal distribution is detailed and the relation to a normal Gaussian distribution is discussed. Thereafter, the results are applied to I_{sub}.

2.1 Confidence Interval of Log-Normal Distribution

In the case of a Gaussian distribution $N(\mu, \sigma)$, the $n\sigma$ distance from μ is frequently used to define the confidence interval. For example, using $\mu \pm 3\sigma$ as endpoints defines the confidence interval that contains all the samples with a probability of 99.7%. For a log-normally distributed random variable, there is no such intuitive correspondence between its standard deviation and the confidence interval, in particular the distribution is not symmetric. In this paper, the following approach is adopted (cf. fig. 1)

1. Any log-normally distributed random variable X with samples x_i is first transformed to a random variable Y using $\log_{10}$: $y_i = \log_{10}(x_i)$.
2. A Gaussian distribution of mean μ and standard deviation σ is fitted to the distribution of the random variable Y.
3. The confidence interval of Y with a confidence level of 99.73% is $[\mu–3\sigma, \mu+3\sigma]$. The corresponding confidence interval of X is

$$X_{\mu\pm3\sigma} = 10^{\mu\pm3\sigma} \quad \text{with } \mu = \frac{\sum \log_{10}(x_i)}{n};\ \sigma^2 = \frac{\sum(\log_{10}(x_i) - \mu)^2}{n-1}. \qquad (2)$$

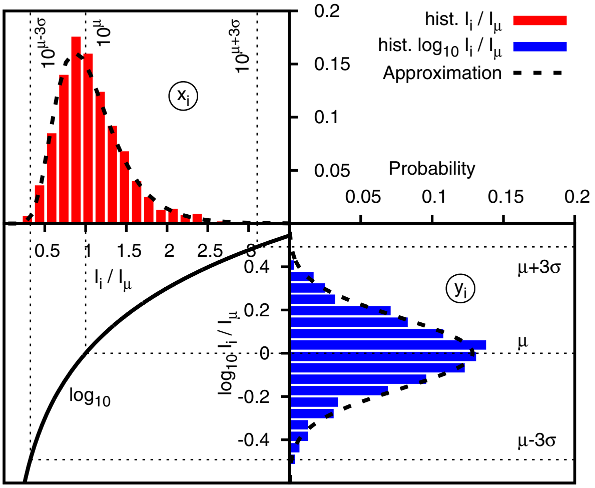


Fig. 1. Derivation of $\mu\pm3\sigma$ confidence interval for MOS transistor currents assuming a Gaussian distribution of transformed current samples (Simulation: MC mismatch with 1k pts, foundry provided models).

2.2 Variability of I_{sub}

In the sub-threshold regime the variability of I_{sub} becomes the main bottleneck in design closure due to its exponential dependency on V_{th}.

On top of random variation such as dopant fluctuations, nanometer process technologies feature significant process spread (wafer-to-wafer variation). Furthermore, variations in propagation delay in super-threshold of -20%...+20% translate to 0.1...10x in sub-threshold. Various techniques have been proposed to compensate for process spread by exploiting the body bias effect (cf. [18]). Although, the effect of well bias on the threshold voltage is decreasing with scaled technologies, it is still feasible to largely compensate for skewed process corners with moderate bias voltages of 300mV in the 40nm CMOS bulk technology used here. Hence, the following analysis focuses on within-die variations (i.e. typical process corner plus mismatch). The normalized $\pm 3\sigma$ confidence interval of currents $ci(3\sigma)$ is

$$ci(3\sigma) = \frac{I_{\mu+3\sigma} - I_{\mu-3\sigma}}{I_\mu} = (10^{-(\mu-3\sigma)} - 10^{-\mu+3\sigma})/10^{\mu} = 10^{+3\sigma} - 10^{-3\sigma}. \quad (3)$$

The normalized confidence interval $ci(3\sigma)$ is shown in fig. 2 as a function of supply voltage for N- and PMOS devices. Also indicated are the sub- and the super-threshold regime. Clearly visible is the considerable widening of the confidence interval for supply voltages below the transistor's threshold. In this technology, the NMOS shows a significantly wider spread as compared to the PMOS device. The remaining analysis will therefore focus on the NMOS.

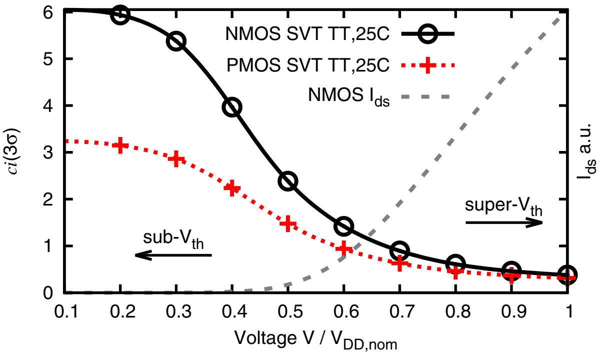


Fig. 2. $ci(3\sigma)$ as a fct. of supply voltage V_{DD} for NMOS and PMOS devices with the minimum channel length and the width equal to 2.5x the minimum size.

Relevant to the width of the confidence interval $ci(3\sigma)$ are changes to the variance σ in (3). As the impact on the variance of V_{th} is linearly scaled with the inverse of the temperature $1/T$ (cf. (1)), it follows

$$\sigma_{V_{\mathrm{th}}}(T_1) = \sigma_{V_{\mathrm{th}}}(T_0)\frac{T_0}{T_1}. \quad (4)$$

Hence, an increase in temperature reduces $ci(3\sigma)$ in the sub-threshold regime (fig. 3, left). Thereby, the curves appear scaled in amplitude and not shifted.

In fig. 3 (right) the confidence interval is plotted for low, regular, and high threshold voltages. The curves appear shifted, i.e. the point when subthreshold operation is reached moves to lower supply voltages for low V_{th} and to higher for high V_{th}. Similar results were observed when using body bias to do V_{th} modulation. In this region proper V_{th} selection can significantly improve variability. Also, the variation $\sigma_{V_{\mathrm{th}}}$ is proportionally improved with higher gate area [15]

$$\sigma_{V_{\mathrm{th}}} \propto 1/\sqrt{W \cdot L} \propto 1/\sqrt{\#\text{fingers}}. \quad (5)$$

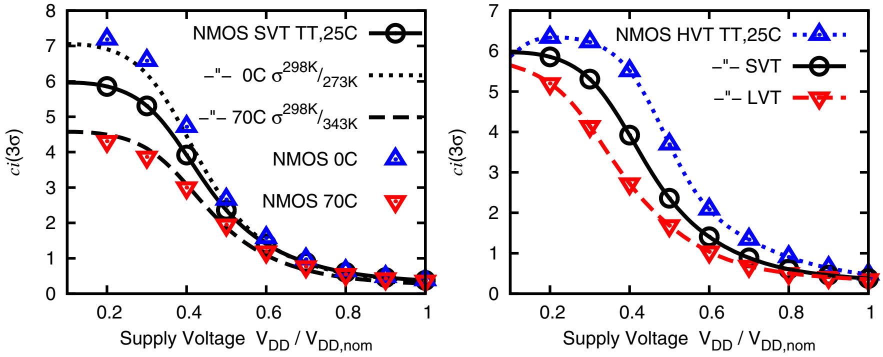


Fig. 3. $ci(3\sigma)$ as fct. of V_{DD} and temp. T (left), and threshold voltage V_{th} (right).

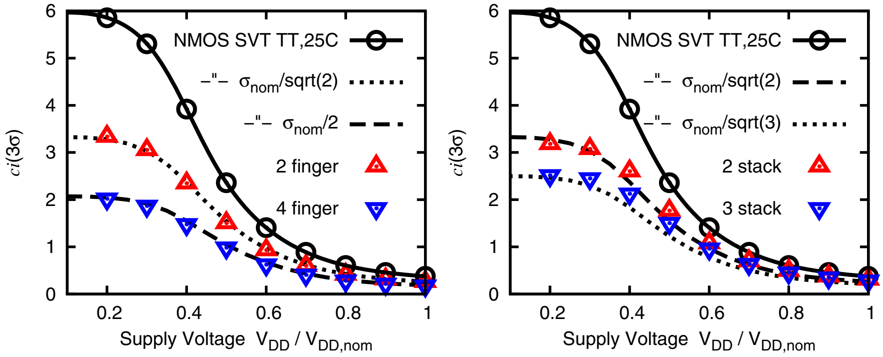


Fig. 4. $ci(3\sigma)$ as fct. of V_{DD} and fingers (left), and transistor stack height (right).

To increase the gate area, we increase the finger count (equivalent to discrete steps in channel width) matching common standard cell methodology. Simulation results of NMOS transistors with different numbers of identical fingers are plotted in fig. 4 (left) together with the trend based on (5). In a similar fashion, the total gate area is increased proportional to the height of the transistor stacks. The effect on the width of the confidence interval is shown in fig. 4 (right). Again, close correlation is achieved applying (5) to (3). The confidence interval *ci* (as fct. of drive and threshold) can be readily modeled across temperature and voltage fitting the parameters of I_{sub} to a base cell and applying correction as detailed above.

3 Derivation of the Noise Margin Model

In the first step, the unity gain noise margin (NM) is computed based on the sub-threshold current model (1). In this paper, the unity gain criterion is adopted [17], which is defined by the unity gain points of the voltage transfer characteristic (VTC) (cf. fig. 5). We define a cell t as failing when its noise margin NM_t falls below the limit L. In the second step, this failure probability per cell F_t is used to predict the reliability of a standard cell library.

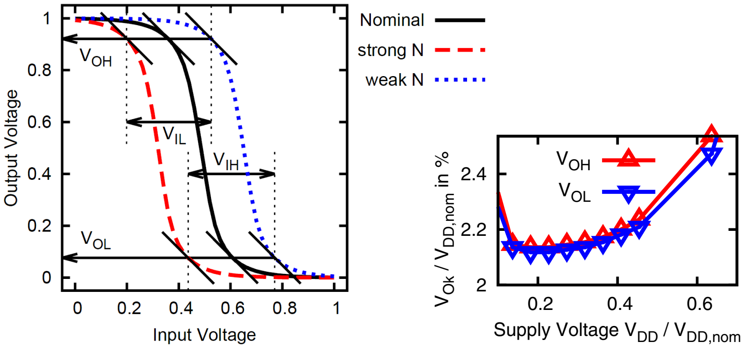


Fig. 5. Normalized voltage transfer characteristic of inverter at V_{DD}=300mV at typical corner and $\pm 3\sigma$ mismatch based on 1k pts MC simulation, and their unity gain points (left). Mean of output voltages $V_{\mathrm{O}k}$ over supply voltage (right).

In sub-threshold the unity-gain Output voltages ($V_{\mathrm{O}k}$ where k is High or Low) are modeled according to [13,14], e.g.

$$V_{\mathrm{OL}} = \frac{n}{2} V_{\mathrm{T}} \quad \text{with} \quad \frac{n}{2} = \frac{n_{\mathrm{n}} p_{\mathrm{p}}}{n_{\mathrm{n}} + p_{\mathrm{p}}}. \tag{6}$$

In particular, equation (6) has no randomly distributed component. Also, there is no voltage or drive strength dependency or random contribution in (6), which matches the simulated output voltage being almost constant (Fig. 5, right) with a standard deviation below 2%. The Input voltages $V_{\mathrm{I}k}$ are derived correspondingly, e.g.

$$V_{\mathrm{IH}} = \frac{n}{2} \left[\frac{V_{\mathrm{DD}}}{n_{\mathrm{p}}} + V_{\mathrm{T}} \left(\ln \frac{\beta_{\mathrm{p}}(n_{\mathrm{p}} - 1)}{\beta_{\mathrm{n}}(n_{\mathrm{n}} - 1)} \right) + \left(\frac{V_{\mathrm{th,n}}}{n_{\mathrm{n}}} + \frac{V_{\mathrm{th,p}}}{n_{\mathrm{p}}} \right) + V_{\mathrm{T}} \ln \frac{2}{n} \right]. \tag{7}$$

Eq. (7) shows the impact of the n:p drive strength ratio $\frac{\beta_{\mathrm{p}}}{\beta_{\mathrm{n}}}$ on the input voltage V_{IH}. Hence, the distribution of threshold V_{th} [14] influences $V_{\mathrm{I}k}$.

Combining the work of [13] and [14] $NM_{\mathrm{H}} = V_{OH} - V_{IH}$ is derived as follows

$$NM_H = V_{\mathrm{DD}} \frac{n_{\mathrm{p}}}{n_{\mathrm{n}} + n_{\mathrm{p}}} - \frac{n}{2} \left[V_{\mathrm{T}} \ln \left(e \frac{2}{n} \frac{\beta_{\mathrm{p}}(n_{\mathrm{p}} - 1)}{\beta_{\mathrm{n}}(n_{\mathrm{n}} - 1)} \right) + \frac{V_{\mathrm{th0,n}}}{n_{\mathrm{n}}} + \frac{V_{\mathrm{th0,p}}}{n_{\mathrm{p}}} \right] + nm. \tag{8}$$

NM_{L} is computed correspondingly by using the absolute value exchanging indices 'n' and 'p' in (8). Here, $V_{\mathrm{th0,n}}$ and $V_{\mathrm{th0,p}}$ are the mean values of the corresponding threshold voltages, and nm is a Gaussian distributed random variable with zero mean and

$$nm = N(0, \sigma_{\mathrm{nm}}^2) \quad \text{with } \sigma_{\mathrm{nm}} = \frac{1}{2} \sqrt{\left(\frac{n}{n_{\mathrm{n}}} \sigma_{V_{\mathrm{th,n}}} \right)^2 + \left(\frac{n}{n_{\mathrm{p}}} \sigma_{V_{\mathrm{th,p}}} \right)^2}. \tag{9}$$

This model predicts an increase in noise margin proportional to the supply voltage. The simulated results of the mean of the noise margin μ_{NM}(fig. 6, left) map

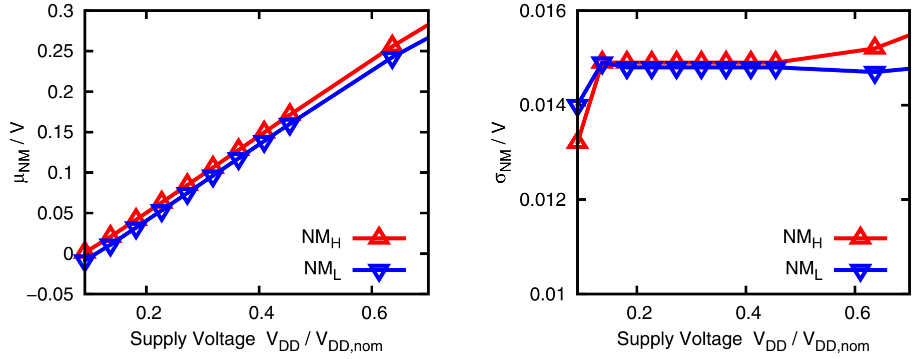


Fig. 6. Mean (left) and standard deviation (right) of NM_k vs. supply voltage

to lines of equivalent slope (N- vs. PMOS has negligible impact) and small offset. The standard deviation is approximately constant in the sub-V_{th}-regime (cf. fig. 6, right).

Next, the probability of a failure F of a cell t is estimated

$$\begin{aligned} F_t &= P(NM_{\text{L},t} \cup NM_{\text{H},t} \le L) = 1 - P(NM_{\text{L},t} > L) \cdot P(NM_{\text{H},t} > L) \\ &= 1 - [1 - P(NM_{\text{L},t} \le L)][1 - P(NM_{\text{H},t} \le L)] \\ &= P(NM_{\text{L},t} \le L) + P(NM_{\text{H},t} \le L) - P(NM_{\text{L},t} \le L \cap NM_{\text{H},t} \le L). \end{aligned} \tag{10}$$

This failure probability is non-Gaussian distribution (cf. fig. 7) due to the minimum selection. For $NM_{\text{L},t}$ and $NM_{\text{H},t}$ being uncorrelated, it reduces to

$$F_t = P(NM_{\text{L},t} \le L) + P(NM_{\text{H},t} \le L) - P(NM_{\text{L},t} \le L) \cdot P(NM_{\text{H},t} \le L). \tag{11}$$

The yield of a cell is $Y_t = 1 - F_t$, which has to be high to achieve good design yield Y. Hence, the product of $P(NM_{\text{L},t} \le L) \cdot P(NM_{\text{H},t} \le L)$ can be neglected leading to the estimate (actually upper limit) of the failure probability (fig. 7)

$$F_t \le P(NM_{\text{L},t} \le L) + P(NM_{\text{H},t} \le L). \tag{12}$$

The actual cell count modulates the limits of the confidence interval of the nm-distribution used in (8). Different cell types are combined by replacing eq (6) with a common worst output voltage of any used cell

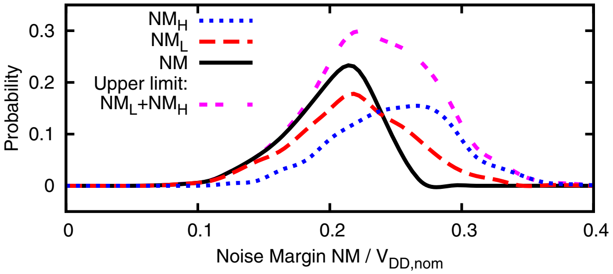


Fig. 7. Simulated NM of inverter (TT,300mV,25C,μ-3σ) and upper limit eq (12)

$$V_{\mathrm{OL}}^{\mathrm{lib}} = \max_t (V_{\mathrm{OL},t}) \quad \text{and} \quad V_{\mathrm{OH}}^{\mathrm{lib}} = \min_t (V_{\mathrm{OH},t}). \tag{13}$$

Finally, the overall yield Y of the design with N different cell types is

$$Y = \prod_t^N Y_t. \tag{14}$$

4 Results and Library Selection

The model of section 3 is applied to a standard cell library in a low-power 40nm CMOS bulk process. As the first step, inverter cells i with different number of identical fingers N_i are compared using the following fitting model based on (8)

$$NM_i = c_0 \cdot V_{\mathrm{DD}} + c_1 + c_2/\sqrt{N_i} \tag{15}$$

with the fitting parameters c_0, c_1, and c_2.

Here, c_0 is a technology parameter in the order of 0.5. Coefficient c_2 determines the variability induced impact of finger count and/or stack height N_i on the noise margin. For the following analysis, the temperature offset and drive mismatch between the complementary CMOS networks are combined in c_1. These parameters are fitted to the results of MC simulation.

Equation (15) is used to predict the scaling of the noise margin as a function of the supply voltage and the finger topology. As an example, fig. 8 (left) highlights the close correlation between the individual simulation results of a specific inverter drive supply voltage combination and the model.

Figure 8 (right) shows a comparison w.r.t. gate types and drive strengths. Increased drive strength largely compensates for mismatch even in the case of unbalanced gates such as ND3 and/or ND4. Hence, neither a specific stack height nor minimal drive strength appears to be an appropriate selection criterion.

A detailed comparison of various gate types (INV, NR, ND) and drive strengths (minimal up to 8x, INV up to 24x) is shown in fig. 9 for individual gates. As anticipated, the noise margin of a cell NM_t improves for higher drive strengths.

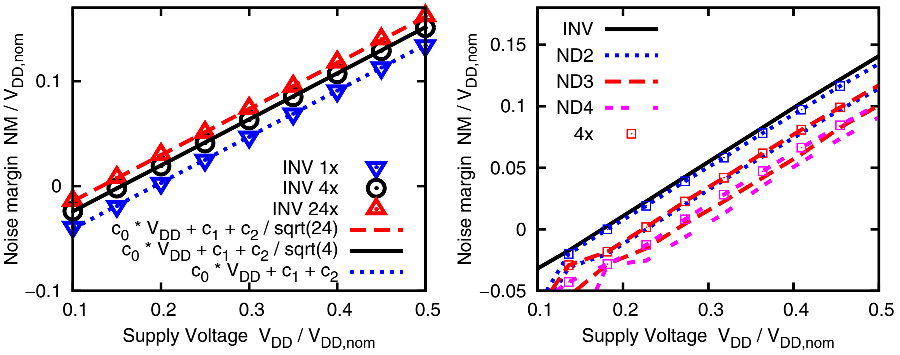


Fig. 8. Model vs. simulation of varying supply voltages and finger counts of inverters (left), and different gate types (2 to 4 input NAND) and 1x vs. 4x drive (right)

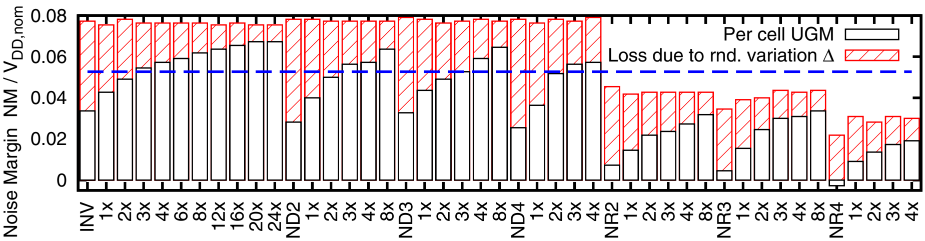


Fig. 9. Noise margin of individual gates (INV, NR, ND) of various drive strengths (minimal to 8x, INV up to 24x). Dashed line in blue highlights noise margin of ND3 3x.

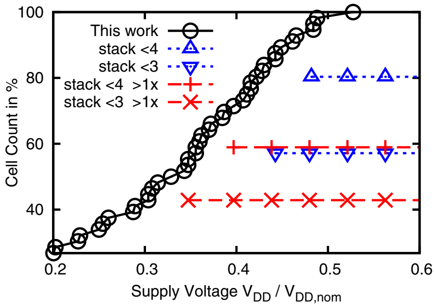
ND-gates feature decreasing *NM* with increasing stack height, which is mainly due to an increase in variability. This is highlighted in the figure as loss due to variability. The full height of each bar indicates the *NM* without variability. On the other side, the NR-gates feature highly degraded noise margin as the imbalance in drive strength of NMOS:PMOS is emphasized by the NR-gate topology (parallel NMOS vs. PMOS stack). Again, it is shown that complex gates of high drive strength can surpass the noise margin of less complex gates of lower drive strength (cf. dashed line in the figure: ND3 3x vs. ND2 or INV).

The following methodology is proposed to select a set of cells T at a given supply voltage that maximizes the noise margin for a reduced cell library:

1. Determine $V_{\mathrm{OH},t}$, $V_{\mathrm{OL},t}$, $V_{\mathrm{IH},t}$, and $V_{\mathrm{IL},t}$ and NM_t for all cells.
2. Start by selecting the cell t with the highest noise margin.
3. Update $V_{\mathrm{O}k}^{\mathrm{T}}$ using (13), all NM_t, and $NM_T = \min_{t \in T} NM_t$.
4. Select the next cell t with smallest impact on NM_T calculated as in 3.
5. Go to 3. until all cells are selected, or NM_T reaches L.

The result of such selection process is shown in fig. 10 for a $\pm 6\sigma$ confidence interval equivalent to a 5M gate design with 99% chip yield. The process can be stopped as soon as a desirable $V_{\mathrm{DD,min}}$ is reached. Also, shown are the results of several brute force selection processes limiting the maximal stack height (less than 4 or 3 transistors) and the drive strengths (more than 1 finger). The proposed selection procedure achieves better results in terms of library size (measured as ratio of selected cell count to total cell count) at the same voltage and lower supply voltage for the same library size. Here, a supply voltage of 580mV would be required to support the full library set at the given design complexity and yield. Here it was assumed, that each cell has the same probability. Exploiting the knowledge of mean and standard deviation of the characteristic voltages and the use of (14) and (15), the flow can easily be adapted to account for different usage of cells in the design, or steer the synthesis to limit the use of cells with low noise margin.

Finally, different figures of merit are summarized in the table of fig. 10 quantifying the advantage of this new methodology over the three best suited brute force selections. At one side, the supply can be reduced while preserving the library complexity. This leads to a supply voltage reduction of up to 70mV achiev-



Library size	Brute force	This work	Diff	Energy Savings
	Min. supply Votlage in mV			
80%	530	464	-66	-23%
59%	437	391	-46	-20%
43%	382	334	-48	-23%

Supply Voltage in mV	Brute force	This work	Diff
	Library size		
530	80%	96%	16%
437	59%	88%	29%
382	43%	55%	13%

Fig. 10. Incremental library cell selection vs. brute force selection

ing energy savings of more than 20%. On the other side, the library complexity is increased by up to 30% at the same minimal supply voltage.

5 Conclusion

The challenges of assuring functionality at low supply voltages have been addressed in two ways. On one side, various simulation and model based analysis have revealed, how techniques such as threshold selection or finger count can be used to reduce variability during cell design. On the other side, a new model has been presented that allows to determine the noise margin or minimal supply voltage of a design as well as enables the selection of library cells to achieve lower voltage or a richer library in the presence of random variations. Applying this model, energy savings of more than 20%, or a library complexity increase of up to 30% can be achieved as demonstrated for a standard cell library in a low-power 40nm CMOS bulk process.

References

1. Wang, A., Calhoun, B.H., Chandrakasan, A.P.: Sub-Threshold Design for Ultra Low-Power Systems. Springer, New York (2006)
2. Wang, A., Chandrakasan, A.P.: A 180mV FFT Processor Using Subthreshold Circuit Techniques. In: ISSCC, pp. 292–293 (February 2004)
3. Kwong, J., et al.: A 65nm Sub-Vt Microcontroller with Integrated SRAM and Switched-Capacitor DC-DC Converter. In: ISSCC, pp. 318–319 (February 2008)
4. Chen, G.K., et al.: Millimeter-scale nearly perpetual sensor system with stacked battery and solar cells. In: ISSCC, pp. 288–289 (February 2010)
5. Jain, S., et al.: A 280mV-to-1.2V Wide-Operating-Range IA-32 Processor in 32nm CMOS. In: IEEE International Solid-State Circuits Conference (ISSCC), pp. 123–125 (February 2012)
6. Seok, M., et al.: A 0.27V 30MHz 17.7nJ/transform 1024-pt complex FFT core with super-pipelining. In: ISSCC, pp. 342–344 (February 2011)
7. Kwong, J., Chandrakasan, A.P.: Variation-Driven Device Sizing for Minimum Energy Sub-threshold Circuits. In: ISLPED, pp. 8–13 (October 2006)

8. Keane, J., et al.: Subthreshold Logical Effort: A Systematic Framework for Optimal Subthreshold Device Sizing. In: DAC, pp. 425–428 (June 2006)
9. Kim, T.H., Eom, H., Keane, J., Kim, C.: Utilizing Reverse Short Channel Effect for Optimal Subthreshold Circuit Design. In: ISLPED, pp. 127–130 (October 2006)
10. Liu, B., Ashouei, M., Huisken, J., Pineda de Gyvez, J.: Standard cell sizing for subthreshold operation. In: DAC, pp. 962–967 (2012)
11. Soeleman, H., Roy, K.: Robust Sub-Threshold Logic for Ultra-Low Power Operation. In: ISLPED, pp. 94–96 (August 1999)
12. Lotze, N., et al.: A 62mV 0.13m CMOS standard-cell-based design technique using Schmitt-trigger logic. In: ISSCC, pp. 340–342 (February 2011)
13. Alioto, M.: Understanding DC Behavior of Subthreshold CMOS Logic Through Closed-Form Analysis. TCAS I 57(7), 1597–1607 (2010)
14. Fuketa, H., et al.: A closed-form expression for estimating minimum operating voltage (VDDmin) of CMOS logic gates. In: DAC, pp. 984–989 (June 2011)
15. Pelgrom, M., Duinmaijer, A.: Matching properties of MOS transistors. In: ESSCIRC, pp. 327–330 (September 1988)
16. De, V., et al.: Techniques for Leakage Power Reduction. In: Chandrakasan, A., et al. (eds.) Design of High-Perf. Microprocessor Circ., pp. 46–62. IEEE Press (2001)
17. Glasser, L., et al.: Design and Analysis of VLSI Circuits. Addison-Wesley (1985)
18. Ono, G., Miyazaki, M.: Threshold-Voltage Balance for Minimum Supply Operation. JSSC 38(5), 830–833 (2003)

TCP Window Based DVFS for Low Power Network Controller SoC

Eyal-Itzhak Nave and Ran Ginosar

Department of Electrical Engineering,
Technion--Israel Institute of Technology,
Haifa 32000, Israel
eyal.nave@intel.com, ran@ee.technion.ac.il

Abstract. The decision to enable a network controller to operate at a high performance mode, at the cost of high power, should not rely solely on the amount of data that needs to be transmitted, but also on the ability of the network to deliver it. This work presents a power reduction approach for network controllers using the TCP protocol's unique capability to sense congested networks. Simulations show that it consistently saves at least 10% more energy than work-load only based DVFS throughout various traffic loads and that it nearly doubles the energy saved at various network congestion levels.

Keywords: DVFS, Low Power, TCP, LAN, Network Controller.

1 Introduction

A 2006 study [1] estimates that, in the U.S. alone, annual energy consumption of networked systems approaches 150 TWh, with an associated cost of around 15 billion dollars. The prevalence of networked mobile devices demands longer battery life and less heat dissipation. Data centers growth struggles with the challenges of cooling data center and lowering electricity costs.

In this study we have developed a novel approach, TCP Window DVFS (TWD), for power saving in the most popular computer networks, those using the TCP protocol. Our network DVFS approach, in contrast with previous approaches, determines the DVFS power mode not only according to the work-load (as may be indicated by accumulated packets in buffers). Rather, we also consider the ability to successfully transfer packets through the network. We use the TCP window size to sense network congestion. That window grows with received acknowledgements and is reduced upon packet loss. We compare this method with a simpler DVFS approach, Packet Buffer DVFS (PBD) [2]. Simulation results show that TWD's energy savings are significantly greater than those of PBD, though both lead to major savings in power consumption.

The rest of this paper is organized as follows: Section 2 surveys previous related work. Section 3 describes the proposed "TCP Window DVFS" (TWD). Section 4 describes the simulation that was used to compare energy savings results. Sections 5

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 83–92, 2013.

and 6 compare energy consumption and saving of TWD across various traffic loads and congestion levels, respectively. Section 7 summarizes this work and offers conclusions.

2 Related Work

In this section we survey previous research aiming to reduce power and energy in networks by modifying protocols of various layers of the OSI model, such as the Data Link Layer (Ethernet) and Transport Layer (TCP).

2.1 Energy Efficient Ethernet

The Energy Efficient Ethernet (EEE) standard (IEEE Std 802.3az-2010) defines mechanisms to stop transmission when there is no data to send. Low Power Idle (LPI) is used instead of the continuous IDLE signal when there is no data to transmit. LPI defines long periods over which no signal is transmitted and short periods when a signal is transmitted to refresh the receiver state to align it with current conditions. [3] shows how packet coalescing can be used to improve the energy efficiency of EEE. EEE is limited to wired network systems using IEEE 802.3 Ethernet protocol. In contrast, our PBD and TWD methods are bounded to neither wired networks nor a specific Data Link Layer protocol, so they can also be applied, for example, to wireless networks. PBD can be utilized in any network where packet buffers are used to store packets before processing. TWD requires, in addition, the usage of TCP as the Transport Layer protocol. EEE is also limited to either LPI mode or full work mode, while PBD and TWD enable multiple DVFS modes for finer tuning of power.

2.2 TCP Level Power Reduction

'Green TCP/IP' has been developed as part of the Energy Efficient Internet Project [4]. It addresses loss of TCP sessions when the CPU is shutting down. Idle hosts are often left fully powered because network protocols and mechanisms fail when the host is not able to conduct basic state-keeping operations. The solution is based on adding a new option in the TCP header ("TCP_SLEEP"), instructing the server to bypass all internal TCP/IP instructions which would drop the connection for that client. Thus, the TCP connection stays open without a need for any activity from the client side (the CPU can shut down). Once the CPU resumes, it can continue sending packets on the open TCP connection without the costly need to reinitiate the TCP connection. However, that solution suffers from three major disadvantages:

1. Energy saving is only achieved when the client is completely shut-down, missing the cases of active idle or low network utilization periods, which comprise a significant part of network activity. Measurements show that the average utilization of desktop Ethernet links is in the range of 1%-5% [5], [6]. Both PBD and TWD provide major energy saving for these low network utilization periods, while maintaining high performance for high utilization bursts/periods.

2. This solution is not adaptive to network conditions. In fact, the “sleep mode” is triggered by a CPU shutdown of the client, regardless of the TCP connection load or network conditions (congestion, link breaks, etc.). Our method is network oriented and adjusts power/performance tradeoff according to TCP connection load and network conditions, taking advantage of data existing in the TCP window.
3. Energy saving is only achieved at the client side. The server side continues to consume energy as if the TCP connections were active in-order to maintain the connections open when clients wake up. In contrast, our solution allows both ends of the TCP connection to use low power mode when possible, thereby enabling double energy saving. Our simulation experiments follow the mutual effects of dynamic power mode changes on both ends of a full duplex TCP connection. Each side includes both RX and TX with independent power managements, comprising a network system with mutual four independent power management systems.

2.3 Power Reduction in Data-Centers and Wide-Area Networks

Data centers are a major source of network energy consumption. The ElasticTree [5], a network-wide power manager, dynamically adjusts the set of active network elements, links and switches, to satisfy changing data center traffic loads. The links and switches that are not needed are turned off. DVFS, as used in our work, enables higher resolution of power management which is not limited to shut-down of a component, but also enables interchanging different work modes.

Chabarek *et al.* [7] use mixed integer programming to optimize router power in a wide area network, by choosing the chassis and line-card configuration to best meet the expected demand. Mandviwalla *et al.* [8] explored using DVS in multi-processor based line-cards. Nedevschi *et al.* [9] investigated network savings with both DFS and DVFS in addition to putting network components to sleep. They propose shaping the traffic into small bursts at edge routers to facilitate putting routers to sleep. Their work compares sleeping vs. rate adaptation in terms of the energy savings achieved across a range of network utilizations. In our work, unlike [7][8][9], DVFS is based not only on the traffic utilization/load, but also on the network congestion and availability.

3 TCP Window Based DVFS

The "TCP window based DVFS" (TWD) policy is based on a simple TCP concept: acknowledgments of transmitted packets indicates that the packets have arrived at their destination and thus the network is able to cope with the current traffic. Failure to receive an acknowledgement results in a sharp decrease of the size of the TCP window because the network may be too congested to successfully deliver the packets of the full window.

Packet Buffer DVFS (PBD) policy [2], as opposed to TWD, determines its power mode based solely on the amount of work to be done, i.e. the size of the packet buffer. When the packet buffer is filled above a threshold, PBD uses high power mode. High power mode maximizes the transmission rate of packets, even when the network is too congested to enable successful delivery of these packets. Such a policy may result in many lost packets which need to be retransmitted. These lost packets cause a decrease in the TCP window's size, which would limit the number of packets transmitted in parallel and would not allow new packets to be transmitted until the transmitted ones are acknowledged.

In such a PBD scenario the LAN controller is in high power mode, but the actual transmission rate is low, limited by the decreased TCP window. Therefore, there is no advantage in using high power mode when the network is congested, and power is wasted. High power mode should only be used when high performance is useful, i.e., when the network is not congested

Fortunately, the same TCP window, which limits the number of transmitted packets when packets start getting lost, can be utilized to sense congestion in the network. With TWD, power mode is determined by both the packet buffer size and the TCP window size. The packet buffer size threshold cannot be ignored, as high power mode is useless and wasteful when there are only a few packets in the packet buffer or it is empty. Lost packets during network congestion cause the TCP window size to decrease. TWD senses this decrease and affects transition to low power mode. Thus the two indicators, packet buffer size and TCP window size, complement each other in low power LAN controller.

4 Simulation Modeling

To compare energy consumption of TWD, PBD and existing baseline non-DVFS network systems, a configurable DVFS simulation environment was developed, simulating different traffic patterns transmitted during a TCP session with different network conditions. We assume separate DVFS work-points (High/Low) and transmission rates for TX and RX. We further assume that the switching time between power states is negligible, but do take into account the energy overhead consumed for the switching. The simulated system is schematically described in Fig. 1.

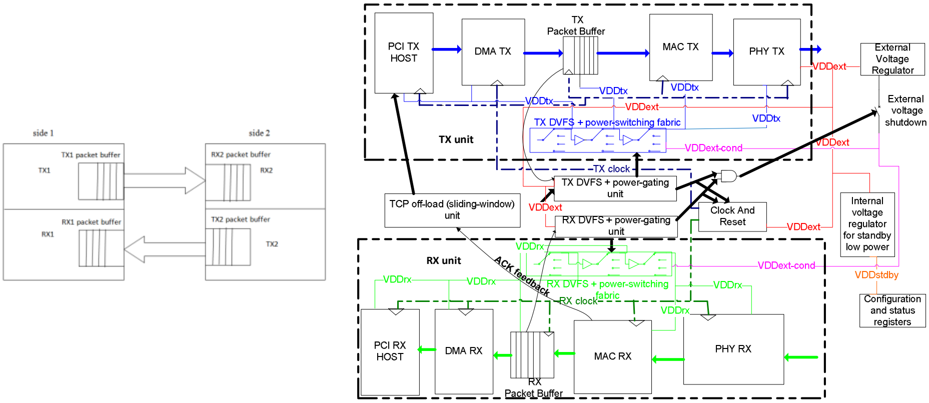

side 1
side 2
TX1 packet buffer
TX1
RX2 packet buffer
RX2
RX1 packet buffer
RX1
TX2 packet buffer
TX2
PCI TX HOST
DMA TX
TX Packet Buffer
MAC TX
PHY TX
External Voltage Regulator
VDDext
VDDtx
External voltage shutdown
TX DVFS + power-switching fabric
TX unit
VDDext-cond
TX clock
TX DVFS + power-gating unit
TCP off-load (sliding-window) unit
RX DVFS + power-gating unit
Clock And Reset
Internal voltage regulator for standby low power
VDDstdby
Configuration and status registers
RX unit
ACK feedback
VDDrx
RX DVFS + power-switching fabric
RX clock
PCI RX HOST
DMA RX
RX Packet Buffer
MAC RX
PHY RX

Fig. 1. (left) simulation architecture scheme: two sides, one port each, separate domains for RX/TX per side; (right) functional clocks and voltage domains of one side

In Fig. 2, simulation plots show TWD transitions to low power mode when sensing network congestion as indicated by a sharp decrease of the TCP window size, as opposed to PBD which stays in high power mode during congestion periods. In addition, TWD transitions to high power mode only when the TCP window size reaches the high threshold, while PBD only requires the packet buffer size to cross its high threshold.

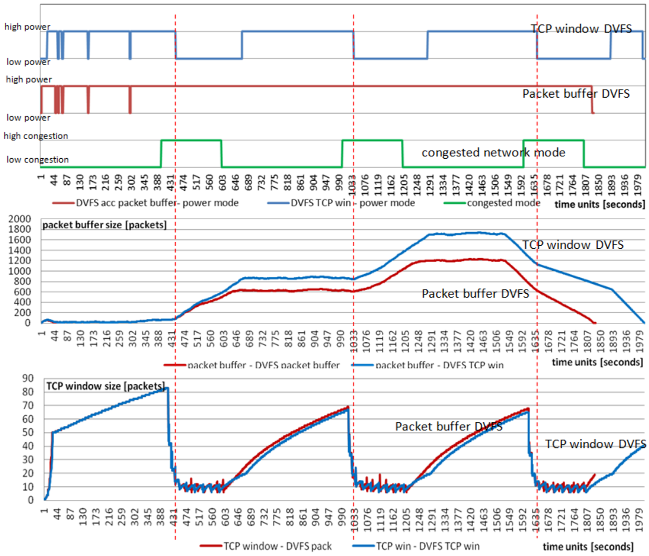


Fig. 2. TCP window DVFS transitions to low power mode when sensing congestion

All simulation runs start with empty packet buffers, and employ a real trace [10] that provides packet arrivals during 1600 seconds. The simulation proceeds beyond 1600 seconds until all packets are delivered and acknowledged.

In the simulation run of Fig. 2, 8,752 and 8,558 packets arrive at the TX of side 1 and side 2 respectively, during 1,600 seconds. The completion time of all these packets and their respective acknowledgements, including lost packets during congested periods, varies across DVFS policy, sides and unit type (TX/RX). The longest simulation run is 2,005 seconds. The network enters a congestion period 400 seconds after the end of the previous congestion period and stays congested for 200 seconds, in which time a packet is lost every 50 seconds.

The total energy consumed by a unit during the simulation run is:

$$E_{unit}=P_{low}*\Delta T_{low} + P_{high}*\Delta T_{high} + E_{lh}*N_{lh} + E_{hl}*N_{hl} \qquad (1)$$

where P_{low} and P_{high} are the power consumption of low and high DVFS modes, respectively, ΔT_{low} and ΔT_{high} are the time the unit has operated in low and high DVFS modes, respectively, and E_{lh} and N_{lh} (E_{hl} and N_{hl}) are the energy overhead and number of transitions from low to high (high to low) DVFS mode, respectively. The total energy of a simulation run is the sum of the energy of all 4 units: TX1, RX1, TX2 and RX2. The energy savings in Sections 5 and 6 are calculated as follows:

$$E_{s(\mathrm{DVFSmodel1\ vs.\ DVFSmodel2})} = E_{\mathrm{total(DVFSmodel2)}} - E_{\mathrm{total(DVFSmodel1)}} \quad (2)$$

where DVFSmodel1,2 are PBD, TWD or No DVFS as appropriate.

5 Comparison of Energy Saving across Traffic Load Levels

DVFS power mode selection depends on packet arrival rate (i.e. the traffic load). Packet arrival rate is modeled as a Poisson distributed stochastic process. The probability that *k* packets will arrive in a single time-unit (Δt=1) is

$$P(k,\lambda)=(\lambda^{k}e^{-\lambda})/k! \quad (3)$$

where λ is the average number of packets arriving per second. Rather than using the trace in [10], we generate packet arrivals during 1600 seconds according to Eq. (3), for a range of rates. The transmission rate is 3λ and λ at high and low power modes, respectively, because typically doubling the controller frequency from 250 to 500MHZ and raising the supply voltage from 1V to 1.2V would triple the controller processing rate.

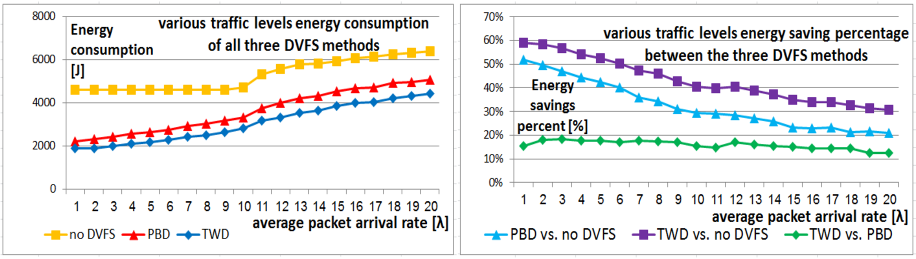


Fig. 3. Different traffic levels: (left) energy consumption; (right) energy saving percentage

Fig. 3 shows the energy consumption and energy saving percentage of the baseline and the two DVFS methods across various packet arrival rates. The network congestion level is held constant for all simulation runs in this section. The time between congested periods is 400 seconds and the length of each congested period is 200 seconds, during which a packet is lost every 50 seconds.

Although packets arrive during 1600 seconds in each simulation run, the completion time varies, as do the time in high power mode vs. time in low power mode and the number of transitions between power modes. According to left graph of Fig. 3, the energy consumption of both PBD and TWD increases with traffic load at an average incremental rate of 158J per unit increase of λ. The maximum energy saving of TWD compared to no DVFS is 2.73KJ, achieved at λ=1. The maximum energy saving of PBD compared to no DVFS is also at λ=1 but is about 300J lower: 2.4KJ. Overall, the energy saving of TWD is higher than PBD by an average of 550J. As can be seen from the right graph of Fig. 3, though the decreasing energy saving trends of PBD and

TWD are the same, an approximately constant gap of more than 10% remains throughout all traffic level loads in favor of TWD.

As the traffic load is increased, the energy saving decreases. This is because high traffic load fills the packet buffer above the high threshold causing the network controller to operate at high power mode and increasing the time percentage that the network controller is in high power mode. This sharp decrease of savings becomes more moderate at about λ=10.

The TCP window's size decreases on every lost packet. Fewer lost packets cause less decrease of the TCP window's size, allowing more packets to be transmitted. If the transmitter is in high power mode but the TCP window's size is small, then energy is wasted since the network controller consumes high power but is not able to transmit as many packets as it could have. In addition, when not using DVFS at high traffic load levels ($\lambda \geq 10$), the performance provided by high power mode is insufficient to constantly keep the buffer below the low threshold, as is the case in low traffic load levels ($\lambda < 10$). This results in energy consumption higher than the roughly constant energy consumption at $\lambda < 10$. However, the energy consumption trends of TWD and PBD remain the same as in the low traffic loads. Therefore, a slight increase in energy saving can be observed when $10<\lambda<15$. In these traffic load levels, many packets are lost due to network congestion. The ability of PBD and TWD to transition to low power mode contributes a significant advantage to less energy consumption.

The energy saving achieved with TWD compared to PBD shows (on the left graph of Fig. 3) an increasing trend as traffic load increases. The relative energy saving at the lowest traffic load (λ=1) is nearly doubled at high traffic load (λ=18), rising from 340J to 700J. When TWD energy consumption is compared to that of PBD, the energy saving percentage is roughly stable, ranging from 12% to 18%. In the low traffic load levels the energy saving percentages are slightly higher with energy saving percentages around 18% , while at high traffic load levels ($\lambda \geq 17$) they are slightly lower around 12%-14%.

TWD transits to low power mode as soon as it senses network congestion, providing two means of energy saving: using lower power and reducing the transmission rate to 1/3 of the high power mode transmission rate, resulting with fewer transmitted packets during a packet loss. The effect of these two advantages is more significant when the number of retransmitted packets is higher, in high traffic load levels. This is why the energy saving of TWD when compared to PBD is higher in high traffic loads. However, as apparent from the right graph of Fig. 3, unlike the actual energy saving that reach their maximum value at higher traffic load levels, the maximum percentage of energy saving happens at low traffic load levels, because the total energy consumption of PBD in high traffic loads is higher.

6 Comparison of Energy Saving across Congestion Levels

We now observe the impact of TWD at various levels of network congestion. Ten congestion levels 1-10 were simulated, where 1 is the least congested network level and 10 is the most congested one. A congested network is characterized by lost packets. The more congested the network is, the more packets are lost. A network is usually not congested 100% of the time. We define simulated congestion levels according to both the frequency and length of the congestion periods and the frequency of packet loss in a congested period. At congestion level 1, the simulation stays 400 seconds in non-congestion mode, and then enters congestion mode and stays there for 200 seconds. In congestion mode, a packet is lost every 200 seconds (once per congestion time period, rather than 4 as in sections 4-5). At congestion level *i*, both the time between congestion periods and the time between lost packets during a congested period are divided by *i*, while the length of the congestion period is multiplied by *i*. As opposed to Section 5, the packet arrival distribution is the same for all simulation runs (extracted from the same real traces [10] as in Section 4). Thus, in this section we isolate the effect of network congestion on the energy saving of each DVFS policy.

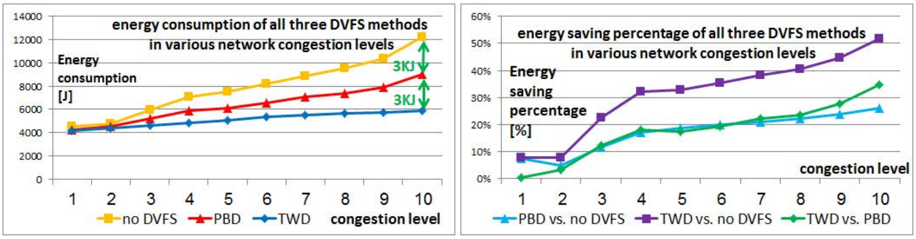


Fig. 4. Different congestion levels: (left) energy consumption; (right) energy saving percentage

According to Fig. 4, at low congestion levels, the energy consumption difference among DVFS modes is small. Level 1 is less severe than congestion in previous chapters. As congestion increases it can be observed that the energy consumption of PBD is about the midpoint between the energy consumption of the baseline and TWD. The energy saving benefit increases with congestion, from less than 1KJ in low congestion levels to more than 3KJ in high levels, in addition to the energy saved with PBD. Clearly, PBD copes well with congestion, and TWD provides even better energy saving. The right graph of Fig. 4 shows that TWD doubles the energy saving of PBD throughout all congestion levels (the blue and green curves are close to each other).

TWD boosts the energy saving percentage in highly congested networks (from 8% to 52%). This is the major benefit from exploiting TCP's ability to sense congested network and react accordingly by transiting to low power mode. Therefore, the TCP window is a better indicator for power mode selection than the packet buffer. As expected, the incremental energy saving in TWD (green graph) increases with congestion. The incremental energy saving of TWD over PBD increases from almost nothing at the lowest congestion level to 35% (3.15KJ). This clearly points out the advantage of using TWD over PBD, especially in high congestion networks.

7 Conclusions

This paper presents a novel approach to power reduction in network controller SoCs, using the advantage of the unique congestion-avoidance feature of TCP to improve previous work-load based DVFS mechanisms. The key idea behind TCP window based DVFS is that the present work-load should not be the only factor for the decision whether to use high-power/high-performance mode or not. Rather, the ability to efficiently carry-out this work must also be considered. For a network controller, the ability to transmit a load of packets depends on the network congestion level. The novelty of this work is in utilizing the TCP window in addition to the packet buffer load for DVFS decision. We have simulated a network to predict the energy savings of this approach over DVFS based only on the size of the packet buffer, and arrived at the following conclusions:

1. TCP window based DVFS achieves higher energy saving than packet buffer based DVFS thanks to its ability to sense network congestion.
2. Both methods of DVFS reach their peak energy saving in low traffic loads. This is important because the main problem with network power reduction is during idle and low traffic periods.
3. The more the network is congested, the more efficient are both DVFS methods.
4. The advantage of TWD compared to PBD in various traffic loads depends on whether the metric is the magnitude of energy saved or the percentage of the energy saved. The magnitude is higher in high traffic load levels because using low power mode during periods of network congestion avoids many energy-wasteful packet retransmissions and packet loss. On the other hand, TWD achieves higher percentage of energy savings compared to PBD in low traffic loads because of the higher total energy consumption consumed at high traffic load levels.
5. TWD achieves higher energy savings compared to PBD in highly congested networks because of its ability to sense the congestion (via the TCP window).

Table 1. Energy saving percentage results summary

DVFS type	Across traffic loads			Across congestion levels		
	min	max	median	min	max	median
PBD vs. No DVFS	21%	52%	29%	7.4%	26%	19%
TWD vs. No DVFS	31%	60%	40%	7.7%	52%	34%
TWD vs. PBD	12%	18%	16%	0.3%	35%	19%

Table 1 summarizes the minimum, maximum and median energy saving percentage results achieved across various traffic loads and various network congestion levels. Comparing row 2 to row 1 proves that TWD indeed provides more energy saving in every criterion and even doubles the max percentage across congestion levels.

Our simulation model uses only two DVFS power modes in-order to simplify the analysis, enabling a clear picture of the benefits of TWD over PBD. Future work may use more complex models, having more power levels and/or usage of Adaptive Voltage and Frequency Scaling (AVFS) to reflect the activity dependency over time.

References

1. Nordman, B.: Networks, Energy, and Energy Efficiency. In: Cisco Green Research Symposium (March 2008)

2. Nave, E., Ginosar, R.: PBD: Packet Buffer DVFS. Technical report (2012), http://webee.technion.ac.il/~ran/papers/NaveGinosarPacketBufferDVFS2012.pdf
3. Christensen, K., Reviriego, P., et al.: IEEE 802.3az: The Road to Energy Efficient Ethernet. IEEE Commun. Mag. (2010)
4. Irish, L., Christensen, K.: A “green TCP/IP” to reduce electricity consumed by computers. In: IEEE Southeastcon, Orlando, FL, pp. 302–305 (April 1998)
5. Heller, B., Seetharaman, S., Mahadevan, P., Yiakoumis, Y., Sharma, P., Banerjee, S., McKeown, N.: Elastictree: Saving energy in data center networks. In: Proceedings of the 7th USENIX Symposium on Networked System Design and Implementation (NSDI), pp. 249–264. ACM (2010)
6. Christensen, K., Gunaratne, C., Nordman, B., George, A.: The Next Frontier for Communications Networks: Power Management. Computer Comm. 27(18), 1758–1770 (2004)
7. Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D., Wright, S.: Power awareness in network design and routing. In: Proc. IEEE INFOCOM (2008)
8. Mandviwalla, M., Tzeng, N.-F.: Energy-Efficient Scheme for Multiprocessor-Based Router Linecards. In: Proceedings of the International Symposium on Applications on Internet, January 23-27, pp. 156–163 (2006), doi:10.1109/SAINT.2006.29
9. Nedevschi, S., Popa, L., Iannaccone, G., Ratnasamy, S., Wetherall, D.: Reducing network energy consumption via sleeping and rate-adaptation. In: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, San Francisco, California, April 16-18, pp. 323–336 (2008)
10. TIPC-over-TCP_disc-publ-inventory_sim-withd.pcap, SampleCaptures - The Wireshark Wiki, http://wiki.wireshark.org/SampleCaptures

Adaptive Synchronization for DVFS Applications

Ghaith Tarawneh* and Alex Yakovlev

School of Electrical and Electronic Engineering, Newcastle University
Newcastle upon Tyne, NE1 7RU, United Kingdom
{ghaith.tarawneh,alex.yakovlev}@ncl.ac.uk

Abstract. We show that synchronizers that operate under Dynamic Voltage Frequency Scaling exhibit exponential failure rate variations due to the disproportionate scaling of propagation delay and the parameter τ. Therefore, the optimum number of synchronization cycles for a design can vary dynamically depending on its voltage/frequency operating point. To address this problem, we present an adaptive clock domain interface that optimizes synchronization latency by evaluating flip-flop synchronization performance dynamically. The proposed design meets a reliability criterion without relying on excessively-conservative synchronizers to accommodate for worst-case performance.

Keywords: Synchronization, metastability, adaptive circuits.

1 Introduction

Modern digital systems consist of large numbers of independently-clocked modules which communicate through asynchronous interfaces. This paradigm overcomes the difficulties of distributing global clock signals and offers lower power consumption and better scalability [1]. Its advantages, however, come at a cost; reliable asynchronous communication between synchronous modules introduces latency and degrades the overall performance of the system.

Latency is incurred because the process of re-timing asynchronous signals according to the receiver's clock is prone to metastability failures [2]. These failures occur when an incoming signal transition violates the setup or hold time conditions of a flip-flop on the receiving domain, causing a prolonged clock-to-q delay and leading to an unpredictable system malfunction [3]. To guard against these failures, a cascade of flip-flops, known as a *synchronizer*, is used to sample the incoming signal before it is used by the receiver's logic. The probability that the synchronizer output will exhibit a prolonged clock-to-q delay is conventionally denoted by its Mean Time Between Failures (MTBF) [3]:

$$\text{MTBF} = \frac{\exp(t_s/\tau)}{T_\text{w} f_\text{c} f_\text{d}} \tag{1}$$

where t_s is synchronization time, τ is a regeneration time constant, T_w is a reference time window, f_c is the clock frequency and f_d is the data arrival rate.

[*] This work was supported by EPSRC grant EP/G066361/1.

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 93–102, 2013.

Given a set of operating constraints (f_c and f_d) and technology-specific flip-flop parameters (T_w and τ), the MTBF criterion of the system is met by allowing a sufficiently-long synchronization time t_s. The latter is commonly chosen as an integer multiple of half the clock period by implementing flip-flop chains of various lengths and triggering edge options. (with the prevalent implementation consisting of two flip-flops, triggered on the same clock edge, to provide one clock cycle of synchronization time). Longer chains provide exponentially better reliability but introduce higher communication latency.

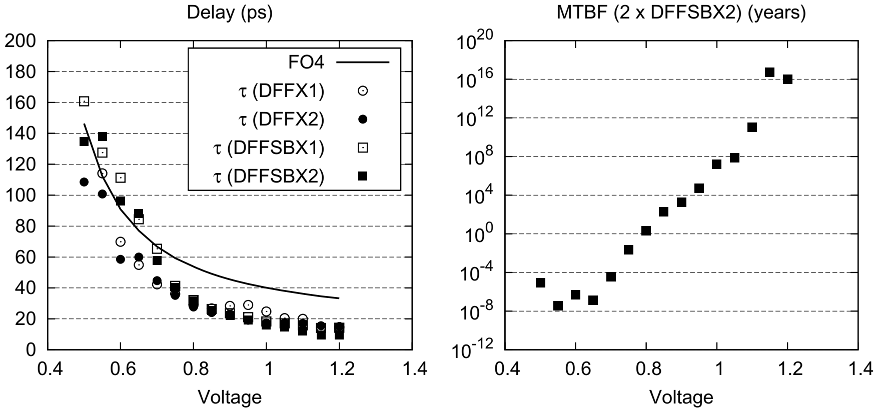
The major difficulty in making the optimal choice in the trade-off between synchronization reliability and latency lies in characterizing the parameter τ. Traditionally, this has been done using elaborate simulation procedures [4], small-signal analysis [5] [6] or on-chip measurements [7] [8]. In practice, however, most designers have no access to transistor-level flip-flop netlists and cannot afford post-fabrication measurements. Therefore, allocating a single clock cycle for synchronization is commonly assumed to be a good trade-off choice.

Synchronizer design is more complex in systems which operate at non-nominal conditions such as lower supply voltages. Under such conditions, the conventional rule of thumb of approximating τ as a fixed fraction of the FO4 delay of the technology has been shown to be inaccurate [5]. Thus, synchronization performance must be re-evaluated on a per-operating-point basis (e.g. subthreshold synchronizer performance is investigated in [9]). Two further complications arise in designs that support Dynamic Voltage and Frequency Scaling (DVFS). First, synchronizer performance does not scale proportionately with changes in voltage and frequency. Second, the scaling of synchronizer performance is highly-dependent on flip-flop design and cannot be predicted without elaborate analysis (e.g. the latches proposed in [5] [10] have better voltage scaling characteristics than typical designs).

This paper has two main contributions. First, we show that synchronizers that operate under DVFS exhibit exponential failure rate variations due to the disproportionate scaling of propagation delay and τ. Therefore, the optimum number of synchronization cycles for the same design can vary depending on its Voltage/Frequency (VF) operating point. Second, we present a clock domain interface that dynamically optimizes the number of synchronization cycles by evaluating flip-flop synchronization performance after each shift in the VF point. Our solution does not require characterizing τ at design time and so it can be used by the wide class of designers who have no access to the simulation tools and transistor-level flip-flop netlists required to characterize τ.

2 Problem Analysis

DVFS is a corner-stone for reducing power consumption when processing highly-variable workloads. Designs that support DVFS continually transition between a number of VF points based on their throughput requirements. These points are typically chosen based on the voltage scaling characteristics of the critical path delay of the design to maximize power savings.



(a) Scaling of τ and FO4

(b) MTBF Degradation

Fig. 1. Impact of VF Scaling on Synchronizer MTBF

VF scaling affects several of the parameters in Equation 1 but its impact on the exponential expression t_s/τ is the most significant. Synchronizing flip-flop chains provide a settling time t_s which is a multiple m of the clock period T. In a DVFS system, T is constrained by the critical path delay of the design at all VF points and so it can be expressed as a fixed multiple n of the FO4 delay of the technology. Thus, synchronization time can be expressed as:

$$t_s = m \times n \times \text{FO4} \tag{2}$$

Therefore, the ratio t_s/τ is, in fact, a design and synchronizer-specific multiple of the ratio FO4/τ and has the same supply voltage dependency. To evaluate how the ratio FO4/τ scales with the supply voltage V, we consider the small-signal models of both the FO4 delay and τ. Assuming square law devices, the FO4 delay can be expressed as [6]:

$$\text{FO4} = \frac{C_L V}{I} \propto \frac{V}{(V - V_{\text{th}})^2} \tag{3}$$

where C_L is the input capacitance of an inverter, I is the drive current of a 4X-smaller inverter and V_{th} is the threshold voltage of the technology.

Similarly, for a cross-coupled inverter pair [6]:

$$\tau = \frac{C_m}{g_m} \propto \frac{1}{(V - 2V_{\text{th}})} \tag{4}$$

where C_m is the bistable node capacitances and g_m is the transconductance of the cross-coupled inverters.

Equations 3 and 4 show that the FO4 delay and τ do not scale proportionately with the supply voltage. The FO4 delay function has a pole at $V = V_{\text{th}}$ while τ has one at $V = 2V_{\text{th}}$. This is because metastability resolution depends on the small-signal characteristics of the latch near the metastable point (roughly $V/2$) while gate transitions occur at the full magnitude of the supply voltage. Therefore, the relative increase of τ at lower supply voltages supersedes that of the FO4 delay leading to a decrease in the ratio t_s/τ. It has been noted in [11] and [12] that the increase in propagation delay at lower voltages compensates for the increase in τ. However, this is true only for supply voltages well above $2V_{\text{th}}$. At lower voltages, synchronizers have exponentially smaller MTBF.

To evaluate the practical severeness of this effect, we used simulation to calculate τ of four flip-flops in a 90nm library and compare it against the FO4 delay. The flip-flops consist of two sizes of a typical data flip-flop DFF and the equivalent sizes of a variant DFFSB which supports asynchronous set. We make two observations from the collected data (Figure 1a). First, the value of τ of all flip-flops increases more significantly than the FO4 delay at lower supply voltages (which supports small-signal analysis). The plot in Figure 1b demonstrates that this effect can reduce the MTBF of a typical 2 flip-flop synchronizer from an extremely conservative figure (10^{16} years) at nominal supply voltage to as low as 1 second at near-threshold voltages. Thus, while one synchronization cycle is sufficient to meet a MTBF criterion of 10^4 years at nominal supply voltage, up to three cycles are required to maintain this figure across the entire supply voltage range. Second, the performance of different flip-flop designs does not scale evenly and so it is difficult to devise a general rule to counteract this degradation.

The disproportionate scaling of the FO4 delay and τ has been investigated in [5] [6] [10] [13] from a technology-scaling perspective and as a performance metric for comparing different latches. However, the impact of this effect on synchronization MTBF in DVFS applications appears not to have been recognized.

3 Proposed Interface

Without being able to characterize τ at design time, long synchronizer chains must be implemented to ensure that the MTBF criterion is met at all VF points. To avoid this conservative strategy, we present a novel adaptive clock domain interface that has four built-in synchronizers of different latencies. The interface senses the ratio FO4/τ dynamically after each shift in the VF point and selects the minimum latency synchronizer that meets the system's MTBF requirement.

3.1 Overview

Our design exploits the fact that satisfying $t_s > R\tau$ (where R is a constant) is sufficient to meet a MTBF criterion without explicit knowledge of either t_s or τ. To illustrate, let P_0 denote the rate at which a synchronizer enters metastable states ($P_0 = T_{\text{w}} f_c f_d$). Equation 1 can now be re-written as:

$$\text{MTBF} = \frac{\exp(t_s/\tau)}{P_0} \tag{5}$$

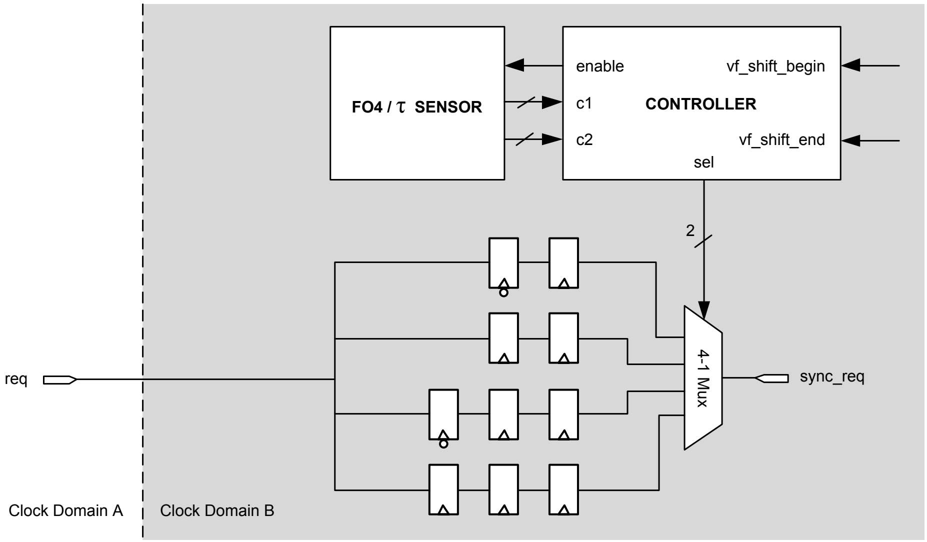


Fig. 2. Proposed Clock Domain Interface

An upper bound on P_0 can be defined as f_d. This is because the number of metastable states encountered during a fixed period of time cannot exceed the number of data synchronization attempts. Now assume that $f_d = 10$ MHz, if $R = 40$ then the MTBF will be 746 years.

Using Equation 2, the Inequality ($t_s > R\tau$) can be expressed as ($m \times n \times \mathrm{FO4} > R\tau$) which can be re-arranged into:

$$\frac{\mathrm{FO4}}{\tau} > \frac{R}{m \times n} \tag{6}$$

Inequality 6 represents a minimum FO4/τ constraint that determines if a synchronizer whose latency is m clock cycles will satisfy the MTBF requirement of the system (R and n are design constants).

The proposed interface, depicted in Figure 2, includes four synchronizers of latencies equal to 0.5, 1, 1.5 and 2 multiples of the clock period (these numbers have been chosen, without any loss of generality, because they are assumed to be sufficient to accommodate latency requirement variations). After each shift in the VF point of the clock domain, the interface uses a built-in sensor to dynamically evaluate the ratio FO4/τ and, using Inequality 6, determine if each of the four synchronizers meets the MTBF requirement of the system. The minimum latency synchronizer from the matching group is then selected and used to synchronize the asynchronous input until the next VF shift.

Table 1 lists the synchronizer selection criteria based on Inequality 6 and the implemented synchronizers ($m = \{0.5, 1, 1.5, 2\}$).

Table 1. Synchronizer Selection Criteria

$\frac{\mathrm{FO4}}{\tau} > \frac{2R}{3n}$	$\frac{\mathrm{FO4}}{\tau} > \frac{R}{n}$	$\frac{\mathrm{FO4}}{\tau} > \frac{2R}{n}$	Required Sync. Cycles (m)	`sel`
1	1	1	0.5	0
1	1	0	1.0	1
1	0	0	1.5	2
0	0	0	2.0	3

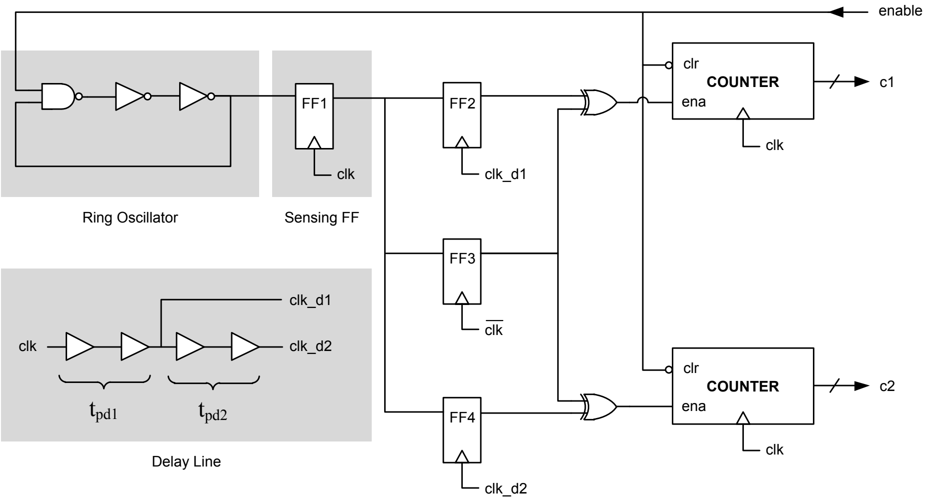


Fig. 3. Schematics of the FO4/τ Sensor

3.2 FO4/τ Sensor

In this subsection, we will demonstrate how the FO4/τ sensor is used to determine if the value of FO4/τ is higher than the thresholds $\frac{2R}{n}$, $\frac{R}{n}$ and $\frac{2}{3}\frac{R}{n}$.

The schematics of the FO4/τ sensor are shown in Figure 3. In principle, this circuit measures the relative increase in the MTBF of a synchronizing flip-flop due to allowing d×FO4 extra time for resolving metastable states (where d is a circuit constant). From Equation 1, increasing t_s by d×FO4 will scale the MTBF by a factor of $\exp(d \times \mathrm{FO4}/\tau)$ which, given the value of d, can be used to calculate FO4/τ.

The circuit consists of a flip-flop FF1 which samples the output of a ring oscillator. Assuming that the output frequency f_{osc} of the oscillator is asynchronous to the sampling clock of FF1 (clk), FF1 will enter metastable states at a rate of

($T_{\mathrm{w}} f_c f_{\mathrm{osc}}$). Two flip-flops FF2 and FF4 sample the output of FF1 at two different times that shortly follow the positive edge of clk. In particular, FF2 samples the output of FF1 after t_{pd1} seconds while FF4 samples it after $t_{\mathrm{pd1}} + t_{\mathrm{pd2}}$ seconds. A much later sample is captured by a fourth flip-flop FF3 at the negative edge of clk.

Due to the occurrence of metastable states, FF1 will exhibit prolonged clock-to-q transitions. We assume that the clock period is long enough such that transitions later than the sampling time of FF3 ($T/2$ seconds after the positive edge of clk) are relatively rare and can be ignored. When a late transition is not captured by FF2, the values of FF2 and FF3 will differ and a counter c1 is incremented. Similarly, when a transition is not captured by FF4, the values of FF4 and FF3 will differ and a counter c2 is incremented.

In essence, the chains FF1-FF2 and FF1-FF4 act as synchronizers whose failures are counted by c1 and c2 respectively. Thus the values of counters c1 and c2 after a fixed period of time t can be derived from Equation 1 as:

$$\mathrm{c1} = t \times T_{\mathrm{w}} f_c f_{\mathrm{osc}} \times \exp(-(t_{\mathrm{pd1}} - t_{\mathrm{cq}})/\tau) \tag{7}$$

$$\mathrm{c2} = t \times T_{\mathrm{w}} f_c f_{\mathrm{osc}} \times \exp\big(-(t_{\mathrm{pd1}} + t_{\mathrm{pd2}} - t_{\mathrm{cq}})/\tau\big) \tag{8}$$

where t_{cq} is the nominal clock-to-q delay of FF1.

From Equations 7 and 8:

$$\frac{\mathrm{c1}}{\mathrm{c2}} = \exp(t_{\mathrm{pd2}}/\tau) \tag{9}$$

Let t_{pd2} represent a pre-determined multiple d of the FO4 delay. Thus:

$$\mathrm{c1} = \mathrm{c2} \times \exp(d \times \mathrm{FO4}/\tau) \tag{10}$$

If the counters are enabled till c2 reaches a pre-defined value, c2 will become a design constant and the only dynamic parameter that will influence c1 will be the ratio FO4/τ. Based on this monotonic relationship, we can pre-determine the values of c1 that correspond to our FO4/τ threshold values $\{\frac{2R}{n}, \frac{R}{n}, \frac{2}{3}\frac{R}{n}\}$ and use them to determine when these thresholds have been crossed. We refer to these corresponding c1 thresholds as $\{k1, k2, k3\}$ respectively and calculate them from Equation 10 as follows:

$$k1 = c2 \times \exp(d \times 2R/n) \tag{11}$$

$$k2 = c2 \times \exp(d \times R/n) \tag{12}$$

$$k3 = c2 \times \exp(d \times 2R/3n) \tag{13}$$

```
while (1)
{
        while (!vf_shift_begin); // wait until VF shift begins

        sel=3;                   // select most conservative synchronizer

        while (!vf_shift_end);   // wait until VF shift ends

        enable=1;                // enable performance sensor

        while (c2!=1024);        // wait until measurement is complete

        enable=0;                // disable performance sensor

        //select optimum synchronizer:
        if      (c1>k1) sel=0;   // FO4 / Tau > (2R/n)
        else if (c1>k2) sel=1;   // FO4 / Tau > (R/n)
        else if (c1>k3) sel=2;   // FO4 / Tau > (2R/3n)
        else            sel=3;
}
```

Listing 1. Controller Implementation

3.3 Controller Behavior

In Subsection 3.1, we have shown that the minimum-latency synchronizer in our interface can be determined by comparing the value of FO4/τ with the pre-computed thresholds $\left\{\frac{2R}{n}, \frac{R}{n}, \frac{2}{3}\frac{R}{n}\right\}$. Following, in Subsection 3.2, we demonstrated that the latter task can be achieved by enabling the FO4/τ sensor till c2 reaches a pre-defined value and then comparing the value of c1 with three corresponding pre-computed thresholds $\{k1, k2, k3\}$. Here, we explain how the interface controller implements the previously described behavior following each shift in the VF point of the clock domain.

Initially, both `vf_shift_begin` and `vf_shift_end` are de-asserted and the controller is idle. When the domain's DVFS controller is about to initiate a change to a new VF point, it asserts `vf_shift_begin`. As soon as `vf_shift_begin` is asserted, the interface controller switches to the most conservative synchronizer (`sel`= 3) immediately. This is necessary because the minimum-latency synchronizer at the new VF point is unknown at this stage and the interface must not permit a MTBF violation under any circumstances.

When the shift is complete, `vf_shift_end` is asserted and the controller enables the FO4/τ sensor by asserting `enable`. The controller then waits for the sensor measurement process to complete (this happens when c2 reaches a pre-defined value, chosen to be 1024 in our design). Subsequently, the value of c1 is compared against the three pre-determined threshold $\{k1, k2, k3\}$ and the lowest-latency synchronizer is selected according to the criteria in Table 1. This behavior is captured by the pseudo-code in Listing 1.

Table 2. Adaptive Interfaces Comparison*

Interface	Latency Control	Area (µm^2)	Power (µW)
[14]†	Fine	625000	1500
Proposed	Coarse	588	61

*cost figures drawn from synthesis in a 90nm technology library
†using a 25k lookup table for log

4 Discussion

4.1 Average Latency

The proposed design uses the most conservative synchronizer during VF shifts and the subsequent FO4/τ measurement process. If these time periods represent a significant fraction of the runtime of the system, the average latency of the interface will be higher than optimum. To mitigate this problem, a lookup-table can be used to store the lowest-latency synchronizer setting after measuring FO4/τ at each VF point. In subsequent shifts to pre-characterized VF points, the optimum synchronizer is selected directly based on the table records.

4.2 Variability

Our interface uses a flip-flop and a delay line to dynamically measure FO4/τ. The design requires that the sensing flip-flop (FF1) has the same τ as the ones in the synchronizer array and that the delay t_{pd2} accurately represents a fraction d of the critical path delay of the system. In practice, these quantities differ due to process variability and so sufficient margins must be allowed when computing the thresholds $\{k1, k2, k3\}$. Allocating these margins to accommodate for component variability will not increase the average latency if the ratio FO4/τ is sufficiently-far from the pre-computed thresholds at all VF points. In all cases, the average latency of the proposed design will be lower than that of a worst-case synchronizer chain.

5 Conclusion

The disproportionate scaling of propagation delay and τ with the supply voltage means that the optimum number of synchronization cycles in a DVFS system can vary depending on the voltage/frequency operating point. Common design flows rely on black-box flip-flop models which do not enable characterizing τ and so it is difficult to mitigate this problem without relying on high-latency synchronizers to accommodate for worst-case performance. We have presented an adaptive interface that can optimize synchronization latency dynamically by evaluating flip-flop synchronization performance after each shift in the operating point. Our design relies on pre-computed thresholds and does not require arithmetic circuits. This makes it more practical than similar adaptive approaches such as [14] where computing the MTBF of synchronization explicitly incurred large area and power overheads (a cost comparison is presented in Table 2).

References

1. Yu, Z., Baas, B.: High performance, energy efficiency, and scalability with gals chip multiprocessors. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 17(1), 66–79 (2009)
2. Chaney, T., Molnar, C.: Anomalous behavior of synchronizer and arbiter circuits. IEEE Transactions on Computers C-22(4), 421–422 (1973)
3. Ginosar, R.: Metastability and synchronizers: A tutorial. IEEE Design Test of Computers 28(5), 23–35 (2011)
4. Yang, S., Greenstreet, M.: Computing synchronizer failure probabilities. In: Design, Automation Test in Europe Conference Exhibition, DATE 2007, pp. 1–6 (April 2007)
5. Yang, S., Jones, I., Greenstreet, M.: Synchronizer performance in deep sub-micron technology. In: 2011 17th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC), pp. 33–42 (April 2011)
6. Portmann, C.L.: Characterization and reduction of metastability errors in CMOS interface circuits. PhD thesis, Stanford University (1995)
7. Kinniment, D., Dike, C., Heron, K., Russell, G., Yakovlev, A.: Measuring deep metastability and its effect on synchronizer performance. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 15(9), 1028–1039 (2007)
8. Semiat, Y., Ginosar, R.: Timing measurements of synchronization circuits. In: Proceedings of the Ninth International Symposium on Asynchronous Circuits and Systems, pp. 68–77 (May 2003)
9. Zhou, J., Ashouei, M., Kinniment, D., Huisken, J., Russell, G., Yakovlev, A.: Subthreshold synchronizer. Microelectronics Journal 42(6), 840–850 (2011)
10. Zhou, J., Kinniment, D., Russell, G., Yakovlev, A.: A robust synchronizer. In: IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures, 2 p. (March 2006)
11. Sakurai, T.: Optimization of cmos arbiter and synchronizer circuits with submicrometer mosfets. IEEE Journal of Solid-State Circuits 23(4), 901–906 (1988)
12. Horstmann, J., Eichel, H., Coates, R.: Metastability behavior of cmos asic flip-flops in theory and test. IEEE Journal of Solid-State Circuits 24(1), 146–157 (1989)
13. Beer, S., Ginosar, R., Priel, M., Dobkin, R., Kolodny, A.: The devolution of synchronizers. In: 2010 IEEE Symposium on Asynchronous Circuits and Systems (ASYNC), pp. 94–103 (May 2010)
14. Zhou, J., Kinniment, D., Russell, G., Yakovlev, A.: Adapting synchronizers to the effects of on chip variability. In: Proceedings of the 2008 14th IEEE International Symposium on Asynchronous Circuits and Systems, ASYNC 2008, pp. 39–47. IEEE Computer Society, Washington, DC (2008)

Muller C-Element Metastability Containment*

Thomas Polzer, Andreas Steininger, and Jakob Lechner

Institute of Computer Engineering, Vienna University of Technology
{tpolzer,steininger,lechner}@ecs.tuwien.ac.at

Abstract. Metastability is the source of many unexpected errors in synchronous circuits. Its mitigation is very well researched in this domain. In contrast, for asynchronous circuits it is normally assumed that the handshaking inhibits metastability. This is, however, only true within the timing closure of the circuit and in the absence of external faults. Metastability may well arise in asynchronous circuits when latching external input signals or when fault tolerance considerations require relaxing the timing closure. Therefore, this paper studies the vulnerability of asynchronous circuits to metastability at the example of a Muller-C element. Traditional mitigation techniques are applied to this kind of circuits and their fitness for Muller-C elements is analyzed.

1 Introduction

An appreciable share of today's circuits is implemented using the synchronous paradigm. However, rising clock speed and increasing parameter variability require the circuit designer to apply large safety margins and employ elaborate algorithms for the clock tree design to guarantee correct operation [1], while still attaining sub-optimal – namely worst case – performance. Additionally, power consumption in general and power dissipation in the clock tree in particular are major concerns, not only in low power applications [2].

Asynchronous circuits are regarded as a very promising alternative. With their local, adaptive timing closure, they can naturally handle parameter variability and always work with average performance instead of worst case [3]. Their dynamic power consumption is lower, and their sub-threshold behavior is much more predictable and scales much better with decreasing supply voltage [4].

While synchronous circuits are well known to suffer from metastability issues at clock domain boundaries [5], it is widely believed that asynchronous circuits only suffer from the so called arbiter problem [6]. The arbiter problem describes the impossibility of deciding a correct sequence of accesses to a single, shared resource, if the requests are not sufficiently separated in time. Other possible sources of metastability are usually disregarded in asynchronous circuit design. Two pronounced technology trends, however, demand for a more intensive consideration of metastability in asynchronous circuits as well: (1) Shrinking feature sizes, albeit being instrumental in boosting performance and power efficiency, make chips more susceptible to single event transients (SETs) caused

* This work is supported by the Austrian Science Foundation (FWF, P21694).

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 103–112, 2013.

by radiation particles. Those SETs do not respect any timing requirements and hence have the potential to infer metastability. (2) The high integration densities featured by modern technology allow the accommodation of complex systems comprising multiple subsystems on a single die. These subsystems often form independent timing domains, at whose interfaces metastability issues inevitably arise, also in asynchronous subsystems. In the second case, an arbiter may be used to resolve metastability, while in the first case this is not possible. However, the arbiter may introduce a significant timing and area overhead.

When designing fault-tolerant asynchronous circuits to counteract (1), the perfect timing closure normally established by handshaking cannot be maintained any more: If a circuit waits for *all* input signals to arrive before proceeding with its operation (as it is normally the case in delay-insensitive asynchronous designs), a lost transition will block the circuit forever. Therefore an exit strategy like time-out or majority vote needs to be applied, which may preempt a late transition on one input [7]. In consequence, that late transition is no more within the timing closure, which introduces the potential for metastability.

So metastability does need attention in the asynchronous domain as well. With this motivation we will investigate the Muller C-element, a basic building block of asynchronous circuits. Our focus will be on options for metastability containment within a single element.

In Section 2 we review the current state of metastability research while in Section 3 we introduce function and implementation of the Muller C-element. Section 4 discusses how metastability manifests itself within a Muller C-element, while in Section 5 we investigate how it may be contained. Section 6 concludes the paper and gives an outlook onto our future research questions in this field.

2 State of the Art

Metastability is the inability to decide between two equally good discrete choices within bounded time. It occurs whenever a mapping from a continuous space is made to a discrete one. In a D-flip flop metastability issues emerge when it comes to judging whether a transition at the input actually occurred before the active clock edge, since data transition and clock transition can be arbitrarily close together in the continuous time space. To avoid metastability, data transitions are simply prohibited by specification within the setup/hold window around the active clock edge. It is easy to imagine that comparable issues apply to all other types of state-holding elements like latches, or Muller C-elements.

From an electrical point of view, in case of a violation of the setup/hold window, the storage loop representing the state of the element is caused to perform a state change but then, half way of doing so, left in an undefined state. The time it takes to decide whether the state will actually flip or return to its previous value is unbounded and impossible to predict [8]. During the decision process, the inner loop nodes are not at well defined digital levels but they take on an intermediate, analog voltage [9]. This voltage can be determined using a Spice DC analysis where the characteristic of the forward and the backward path of the loop, in the simplest case two inverters, are overlaid according to the rules of graphical network analysis. Figure 3(a) shows an example. Beyond the two stable intersection points representing low and high (top left and bottom right), there is an unstable one in the center called the metastable point [10].

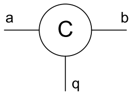
Metastability is a very well researched phenomenon, at least for D-latches and D-flip flops. Mentioned already in 1966 by I. Catt [10], the phenomenon was investigated over the years using analytic models (e.g. for predicting the failure rates [11]), measurements [9,12], and Spice simulations [13]. The latter provide either the metastable voltage levels (DC analysis) or voltage traces of nodes within the storage loop over time (transient analysis [13]). Especially transient analysis may take a long time, as different overlaps of the input signals must be simulated until the metastable state is found. To speed up this process the concept of bisection was developed [14].

With respect to asynchronous circuits, there is little related work available. The behavior of mutual exclusion elements was thoroughly investigated, e.g. in [6]. These elements are built based on an RS-latch where R and S serve as request inputs. Only one of its grant outputs is allowed to be high at any given time. Unfortunately, when both requests are asserted nearly at the same time, its outputs may become metastable. Therefore the decision which request to grant may take an arbitrarily long time. During this time interval both outputs may stay at an intermediate voltage level. To avoid ambiguous interpretation by successor stages, this must be suppressed. This can be achieved by using a low threshold inverter or a so called metastability filter [6,15] as output stage to hide this intermediate voltage. A more detailed description of the functionality of low threshold inverters will be given in Section 5.1.

As the low threshold inverter only works for low-high transitions, one may consider a Schmitt-Trigger for circumventing metastability in the general case. Its hysteresis will shift the threshold accordingly for both transition types and therefore can be used to contain the analog voltage levels of the internal nodes within the storage loop. The usage of Schmitt-Triggers to suppress metastable outputs of latches for building a merge element was already discussed in [16], however, it is not universal. For synchronizers, e.g., the delay additionally introduced tends to be counterproductive and may even increase the upset rate [8]. Metastability mitigation using Schmitt-Triggers for an asynchronous clocking circuit was presented in [17]. [18] on the other hand shows how metastability propagates through an elastic pipeline.

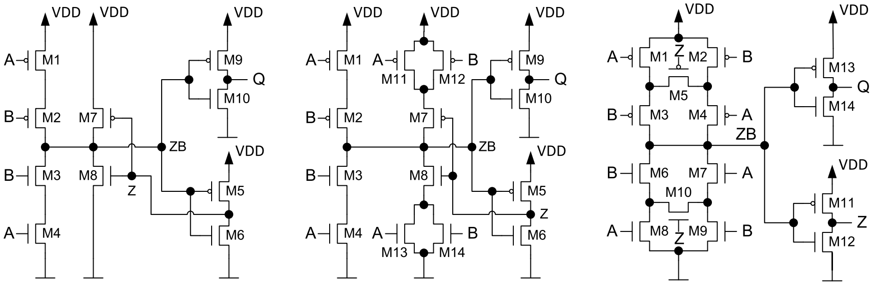
3 Muller C-Elements and Asynchronous Circuits

Muller C-elements [19] (MCEs) are the basic building blocks of the control paths in most asynchronous circuits [20,21]. Basically a MCE is a conjunction for signal transitions: Its output will only go high, if a rising edge was detected on both of its inputs and will only go low, if a falling edge has occurred on both. The truth table for the MCE and its circuit symbol are shown in Figure 1. Note that in case of non-matching inputs the previous output is retained, which causes the need for storage capability. This in turn creates a potential for metastability.



a	b	q_{n+1}
0	0	0
0	1	q_n
1	0	q_n
1	1	1

Fig. 1. Schematic symbol and truth table of a Muller C-element



(a) Weak-feedback (b) Conventional (c) van-Berkel

Fig. 2. Muller C-element implementation circuits

There are several ways of implementing a MCE in CMOS. The three most important variants are the *weak feedback-implementation*, the *conventional implementation* and the *van-Berkel implementation*. These are shown in Figure 2, for details and transistor sizing rules see [22].

4 Metastability Behavior of Muller C-Elements

As all these MCE implementations comprise a storage loop, setup and hold constraints need to be respected for a proper function, just like for a flip flop. In particular short phases of matching inputs, both contradicting the output, must be avoided as this represents the borderline case between clear "switch" and clear "hold". The most relevant respective scenarios both start in the state of non-matching inputs, than (a) a glitch at one input occurs or (b) both inputs change nearly at the same time. This violation of the timing constraints will result in an increased output delay. If the output signal is read before that delay has elapsed, a metastable upset has occurred. This involves the risk of reading an ambiguous intermediate voltage.

Unfortunately metastability is able to overcome conventional error containment boundaries. Therefore all reasoning and all proofs on the correctness of a circuit may become void in its presence. The root for this behavior is that all proofs are based on the specified functional description of logic gates. Metastability on the other hand causes out-of-specification operation of these gates and is hence normally ignored in these descriptions. Therefore it is very important to understand how metastability affects different kinds of circuit elements. This knowledge is the foundation for devising a functioning containment mechanism.

In this paper we concentrate on MCEs. As these are asynchronous components their behavior is quite different from synchronous elements like flip flops. This is mainly due to the fact that the elements do not have the same temporal masking as in the case of flip flops where only active clock edges must be considered. We have modeled the three standard implementation circuits from Section 3 in hSpice using transistors of an industrial 90nm technology library.

Our simulations were done for a single transistor sizing (one according to [22]) and are intended as a showcase for our approach. Although the results will quantitatively depend on the sizing, the qualitative message holds for all practical ones.

4.1 Simulation Algorithm

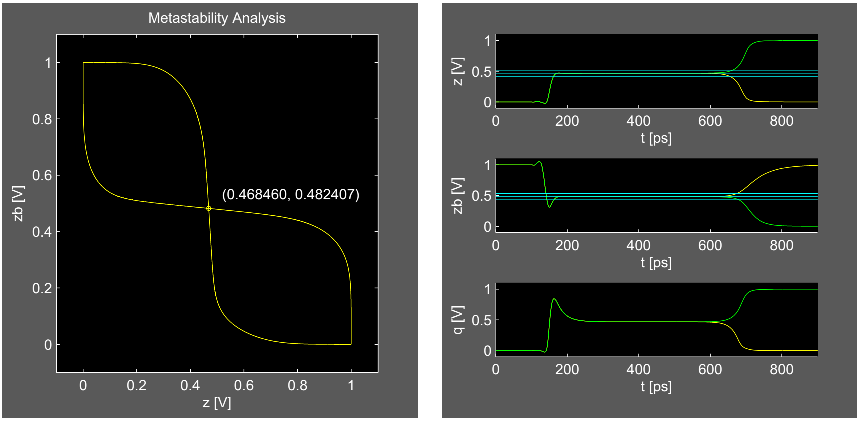
The MCE models were subjected to a Spice DC analysis to determine the metastable voltage values for the inner nodes of the feedback loop. By simulating the forward and backward paths of the loop separately and overlaying them afterwards using MATLAB, we were able to extract the metastable point (recall Section 2) from the resulting plots.

A Spice timing analysis was then used to determine the metastable response over time. We created voltage traces of the storage loop as well as for the output nodes. We drove the MCE into metastability by overlapping matching input signals that would require the internal state to flip for a very short time. The exact duration of the required overlap was extracted using a binary search on the input interval. We implemented the binary search in MATLAB and used hSpice as simulation back-end. When the binary search was finished, the upper and lower bound traces were plotted and used to analyze the metastability behavior.

4.2 Simulation Results

The simulations were done for all three MCE implementations. The resulting trace for the van Berkel implementation is shown in Figure 3. The figure confirms that the inner nodes (Z, ZB) of all implementations go to a non digital level while the element is metastable. The straight lines in the figure represent the result of the DC analysis and an interval of $\pm\frac{V_{DD}}{20}$ around this value. It is apparent that the analog voltage level found in the transient analysis complies nicely with the result of the DC analysis. As the results for the other implementations vary only in the length of the metastable state, they were omitted here but can be found at `http://www.vmars.tuwien.ac.at/documents/extern/3060/patmosappendix`.

It is also visible in the figures that the analog voltage level propagates from the inner node to the output of the MCE. This is because in our simulations we had chosen the same threshold voltages for all inverters, and thus the metastable voltage matches the threshold voltage of the output inverter quite well. With respect to the aspired metastability containment this is not satisfactory. Since it has been proven impossible to prevent metastability in the first place [23], we at least want to confine the metastable voltage to the inner node to mitigate its propagation. We will therefore continue with analyzing the behavior of the



(a) Metastable voltage

(b) Signal trace

Fig. 3. Simulation result for a van Berkel Muller C-element

output inverter and try to redesign it so as to contain metastability within the element. We base our efforts on the techniques already in use for other elements (see Section 2) and will analyze which of these techniques may lead to a better metastability containment of the MCE.

As only the existence of an analog voltage level for an extended time and not the exact length of the metastability is of concern for our analysis, we focus our efforts on the van-Berkel implementation in the following.

5 Metastability Containment

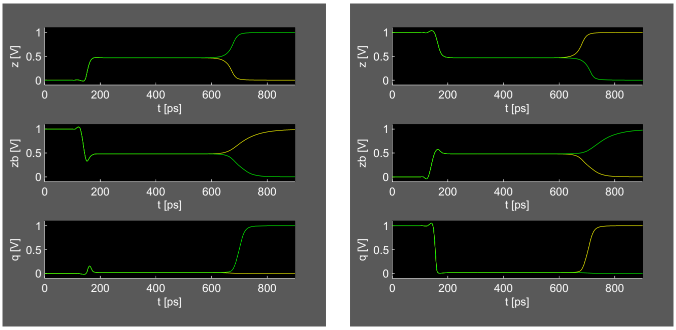
As the metastable voltage can be efficiently determined by a DC analysis in advance, it is possible to devise the metastability counter-measures that are based on its knowledge. More specifically we can select the threshold voltage of the output inverter in such a way that the metastable voltage is uniquely regarded as a high or low value. In this way the output inverter will safely convert the intermediate voltage level into a defined low or high value at the MCE's output. This is the idea behind the approaches presented in this section.

5.1 Low- and High-Threshold Inverters

As already mentioned, for mutual exclusion elements low-threshold output inverters are used to delay the output for the low-high transition[1] until the RS-latch has left the metastable area [6]. When using the solution for the MCE, the low-high transition works as expected and is delayed until the metastable state

[1] This may be counter-intuitive but because the output is inverted, the threshold must be on the same side of the metastable voltage as the start value of the output (therefore the input value of the inverter will be on the opposite side).

has been resolved (see Figure 4(a)). Unfortunately for the high-low transition a glitch is created: When the element is entering the metastable state, the voltage level at the internal node is crossing the threshold and the output switches from high to low immediately. If the metastability then resolves to the original state of the element, the voltage on the internal node again crosses the threshold and the output signal switches back to high (see Figure 4(b)). This glitch in the high-low transition is not a problem in the mutual exclusion element, since the critical phase of the arbitration process always starts in the low state. That is why this approach works fine there.



(a) low-high (b) high-low

Fig. 4. Simulation result for the low threshold inverter

We expect from the containment circuit, however, that the state is cleanly switched in both directions. Especially in asynchronous control circuits the above mentioned glitch may propagate and be the cause for unexpected behavior.

When using a high-threshold inverter instead, a dual problem can be observed. In this case the high-low transition works as expected but the low-high transition creates a pulse. In general we can distinguish four cases that must be considered when designing the containment circuit. Table 1 shows how the different cases are handled by the measures shown in this paper.

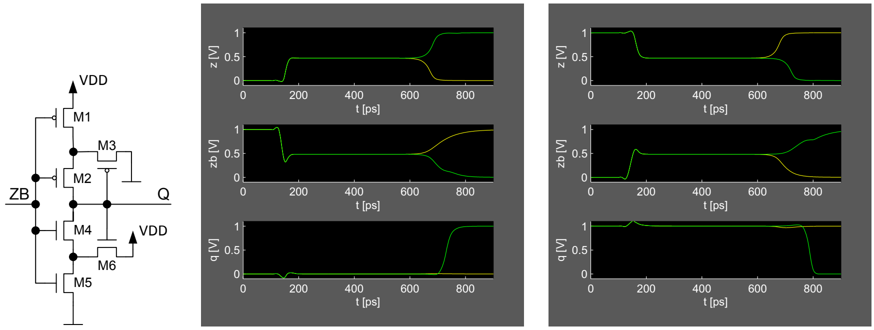
Based on these observations it becomes clear that an element with an adaptive threshold is needed to contain metastability within the element. The threshold must be raised when the output is high and lowered when the output is low, so that it is above the metastable voltage for high-low and below the metastable voltage for low-high transitions, respectively. The natural choice for such an element is an inverting Schmitt-Trigger.

Table 1. Comparison of the metastability mitigation implementations

	low → high	high → low	low → low	high → high
Low threshold inverter	OK (late)	OK	OK (suppressed)	glitch
High threshold inverter	OK	OK (late)	glitch	OK (suppressed)
Schmitt-Trigger	OK (late)	OK (late)	OK (suppressed)	OK (suppressed)

5.2 Schmitt-Trigger

Figure 5 confirms that when using a Schmitt-Trigger as output stage of the MCE, both the low-high and high-low transition work properly. Due to the hysteresis, the switching threshold of the Schmitt-Trigger is adapted in such a way that for both transitions just moving to the metastable state is not sufficient for the analog voltage to reach the threshold, and therefore the output signal does not change its state. Only if the element definitely changes its state, the threshold is crossed and the output signal switches as well.



(a) Circuit (b) low-high (c) high-low

Fig. 5. Circuit and simulation result for the Schmitt-Trigger

Notice that this result is essentially different from what [8] claimed for synchronous circuits: There the additional delay introduced by the Schmitt-Trigger consumes so much of the available resolution time that the overall upset probability is even increased. The essential problem here is that in synchronous circuits the resolution time for metastability is globally determined by the fixed clock frequency in an open-loop fashion [8]. Quite in contrast to that, in the asynchronous case the resolution time is naturally extended as required, as the handshaking forms a closed loop timing control. In this way the output can be delayed without increasing the upset rate. The only constraint is a clean transition from one output level to the other without staying at an intermediate voltage and without glitches – and this is what the Schmitt-Trigger can effectively provide.

After having shown the applicability of the Schmitt-Trigger approach for metastability containment in MCEs, let us briefly analyze its penalties:

1. *Area Overhead:* The standard van-Berkel MCE implementation consists of fourteen transistors (including the output inverter). The proposed version with the Schmitt-Trigger output stage requires four additional transistors. This yields an area overhead of 28.57%.
2. *Delay Penalty:* The nominal output delay of the standard MCE in the used technology is 28 ps. The Schmitt-Trigger increases this delay to 60 ps (214%), leading to an overhead of 32 ps in the fault-free case.

While the area overhead is quite moderate, the delay penalty is considerable. However, protection against metastability has its price in the synchronous domain, as well. At timing domain boundaries this reduction in performance will limit the sustainable data transfer rate.

6 Conclusion and Future Work

We have motivated the need to consider metastable upsets in asynchronous circuits, other than arbiters, as well, as these may occur due to transient faults, due to relaxed timing closure in fault tolerance schemes, or simply at timing domain boundaries. We have consequently analyzed the metastability behavior of asynchronous circuit components at the example of a Muller C-element. Our analysis has shown that all common implementations may become metastable and have the same basic behavior – they potentially exhibit an intermediate voltage. This voltage can be determined in a DC analysis.

Unlike in case of the arbiter, the MCE requires a safe metastability resolution for both output transitions, hence a high- or low-threshold inverter does not suffice, as we have shown that it produces glitches. In contrast to synchronous designs, in turn, resolution time is flexible in delay insensitive circuits, hence a Schmitt-Trigger output stage can be applied, and our simulations confirmed that it indeed works very effectively in containing metastability within the element.

This result only relies on the ability to determine the metastable voltage of the storage cell. The occurrence of such a voltage level is independent of transistor sizing and the actual MCE implementation.

In our follow up work we plan to implement a test chip for “real-world” measurements of generation and propagation of metastability. This will enable us to verify and calibrate our simulations on real silicon.

References

1. Friedman, E.G.: Clock Distribution Networks in Synchronous Digital Integrated Circuits. Proceedings of the IEEE 89(5), 665–692 (2001)
2. Naffziger, S.D., Colon-Bonet, G., Fischer, T., Riedlinger, R., Sullivan, T.J., Grutkowski, T.: The Implementation of the Itanium 2 Microprocessor. IEEE Journal of Solid-State Circuits 37(11), 1448–1460 (2002)
3. Nowick, S.M., Singh, M.: High-Performance Asynchronous Pipelines: An Overview. IEEE Design and Test of Computers 28(5), 8–22 (2011)
4. Chang, I.J., Park, S.P., Roy, K.: Exploring Asynchronous Design Techniques for Process-Tolerant and Energy-Efficient Subthreshold Operation. IEEE Journal of Solid-State Circuits 45(2), 401–410 (2010)
5. Ginosar, R.: Fourteen Ways to Fool Your Synchronizer. In: International Symposium on Asynchronous Circuits and Systems, pp. 89–96 (2003)
6. Kinniment, D.: Synchronization and Arbitration in Digital Systems. Wiley (2007)
7. Cheng, F.-C., Ho, S.-L.: Efficient Systematic Error-Correcting Codes for Semi-Delay-Insensitive Data Transmission. In: International Conference on Computer Design, pp. 24–29 (2001)

8. Kleeman, L., Cantoni, A.: Metastable Behavior in Digital Systems. IEEE Design and Test of Computers 4(6), 4–19 (1987)
9. Chaney, T.J., Molnar, C.E.: Anomalous Behavior of Synchronizer and Arbiter Circuits. IEEE Transactions on Computers C 22(4), 421–422 (1973)
10. Catt, I.: Time Loss Through Gating of Asynchronous Logic Signal Pulses. IEEE Transactions on Electronic Computers EC-15(1), 108–111 (1966)
11. Veendrick, H.: The Behavior of Flip-Flops Used as Synchronizers and Prediction of Their Failure Rate. IEEE Journal of Solid-State Circuits 15(2), 169–176 (1980)
12. Zhou, J., Kinniment, D.J., Dike, C.E., Russell, G., Yakovlev, A.: On-Chip Measurement of Deep Metastability in Synchronizers. IEEE Journal of Solid-State Circuits 43(2), 550–557 (2008)
13. Kacprzak, T., Albicki, A.: Analysis of Metastable Operation in RS CMOS Flip-Flops. IEEE Journal of Solid-State Circuits 22(1), 57–64 (1987)
14. Yang, S., Greenstreet, M.R.: Simulating Improbable Events. In: ACM/IEEE Design Automation Conference, pp. 154–157 (2007)
15. Kinniment, D.J., Bystrov, A., Yakovlev, A.: Synchronization Circuit Performance. IEEE Journal of Solid-State Circuits 37(2), 202–209 (2002)
16. Greenstreet, M.R.: Real-Time Merging. In: International Symposium on Advanced Research in Asynchronous Circuits and Systems, pp. 186–198 (1999)
17. Nystroem, M., Martin, A.J.: Crossing the Synchronous-Asynchronous Divide. In: Workshop on Complexity-Effective Design (2002)
18. Fuchs, G., Fuegger, M., Steininger, A.: On the Threat of Metastability in an Asynchronous Fault-Tolerant Clock Generation Scheme. In: IEEE Symposium on Asynchronous Circuits and Systems, pp. 127–136 (2009)
19. Mead, C., Conway, L.: Introduction to VLSI Systems. Addison-Wesley (1979)
20. Sparso, J., Furber, S.: Principles of Asynchronous Circuit Design - A Systems Perspective. Kluwer Academic Publishers (2001)
21. Sutherland, I.E.: Micropipelines. Communications of the ACM 32(6), 720–738 (1989)
22. Shams, M., Ebergen, J.C., Elmasry, M.I.: A Comparison of CMOS Implementations of an Asynchronous Circuits Primitive: the C-Element. In: International Symposium on Low Power Electronics and Design, pp. 93–96 (1996)
23. Kleeman, L., Cantoni, A.: On the Unavoidability of Metastable Behavior in Digital Systems. IEEE Transactions on Computers C-36(1), 109–112 (1987)

Low Power Implementation of Trivium Stream Cipher

J.M. Mora-Gutiérrez[1], C.J. Jiménez-Fernández [2], and M. Valencia-Barrero[2]

[1] Instituto de Microelectrónica de Sevilla,
Centro Nacional de Microelectrónica (CSIC). Sevilla, España,
[2] Instituto de Microelectrónica de Sevilla /Universidad de Sevilla. Sevilla, España

Abstract. This paper describes a low power hardware implementation of the Trivium stream cipher based on shift register parallelization techniques. The design was simulated with Modelsim, and synthesized with Synopsys in three CMOS technologies with different gate lengths: 180nm, 130nm and 90 nm. The aim of this paper is to evaluate the suitability of this technique and compare the power consumption and the core area of the low power and standard implementations. The results show that the application of the technique reduces power consumption by more than 20% with only a slight penalty in area and operating frequency.

1 Introduction

Cryptography provides techniques, mechanisms and tools for secure private communication and authentication on Internet and other open networks. It is almost certain that in the coming years every bit of information flowing through a network of any kind will have to be encrypted and decrypted. All devices connected to a network should therefore incorporate mechanisms that implement cryptographic functions to ensure safe transfers. With this in mind, it is necessary to design and implement hardware structures which are suitably efficient in terms of area, operating frequency and power consumption. An additional challenge is that implementations must be constructed to withstand cryptographic attacks launched against them by adversaries who have access to both primary channels (communication) and secondary channels (power consumption, electromagnetic radiation, etc.).

Cryptographic algorithms are either symmetrical - those based on the existence of a secret key - or asymmetric - those based on the existence of pairs of public and private keys. Both types play important roles in current applications. Public key-based cryptography was invented by Diffie and Hellman in 1976 [1]. Asymmetric algorithms use a different key to encrypt and decrypt (a public key and private key). The public key can be known to all, while the private key is known only by the receiver of the message. The message is encrypted using the public key and can only be decrypted using the corresponding private key. With this type of cryptography there are no problems with key distribution, but the high complexity of the algorithms requires large computing resources and hardware solutions are complex and involve high power consumption.

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 113–120, 2013.

For private key-based cryptography, the key for encrypting and decrypting is the same and it is secret. Private key-based algorithms can be divided into two groups: block ciphers and stream ciphers. Block ciphers encrypt blocks of data of fixed length while stream ciphers encrypt an amount of data of arbitrary length. These types of algorithms are problematic in that the keys must be distributed between the sender and the receiver. However, the advantage they are faster and their implementations are simple, so they are used to encrypt and to decrypt large amounts of data, as in applications requiring low-complexity algorithms (for example short range wireless encryptions). From the hardware point of view, the resulting implementations are of very low complexity. This makes them ideal for portable devices with low computing capacity and very high power consumption restrictions.

These needs led the European Union to launch an initiative known as eSTREAM [2][10], proposing new stream ciphers which, in both software and hardware, would meet current needs in the field of stream ciphers. This initiative has now identified and published four new algorithms specially designed for implementation in software (HC-128, Rabbit, Salsa 20/12 and Sosemanuk) and three designed to give optimal performance in hardware (Grain, Mickey and Trivium) [3].

Secure data transfers are also becoming increasingly necessary on devices with low computing capacity and which also require low power consumption. The best option for these devices is to use secret key cryptographic algorithms. We have chosen the Trivium stream cipher of all of the possible algorithms available, because it is endorsed by the eSTREAM project and is simple to implement. Furthermore, power consumption can readily be reduced in its architecture through the technique of logical shift register parallelization. With this technique we have obtained implementations with more than a 20% drop in power consumption and virtually no losing performance.

This paper is organized as follow. Section 2 briefly describes the specification algorithm, Trivium. In section 3 the low power implementation is described, showing the synthesis and power consumption reports and comparing them with the standard version. Finally, Section 5 concludes the paper.

2 Trivium Specification

Trivium is a synchronous stream cipher designed to generate up to 2^{64} bits of key stream from an 80-bit secret key and an 80-bit Initialization Vector (IV). The architecture of this cipher is based on a 288-bit cyclic shift register accompanied by an array of combinational logic (AND, OR and exclusive-or) to provide its feedback [4][10].

Firstly the cipher's 288-bit internal state is initialized using the secret key and the initial value, which are loaded into the internal state register. The state is then updated four times 288 without producing key stream bits. This is summarized in the VHDL code below:

```
state(92 downto 0) <= X"0"  & key; -- first register
state1(76 downto 93) <= X"0" & iv; -- second register
state(287 downto 177) <= "111" & X"0"; -- third register
state(92 downto 0) <= state(91 downto 0) & t3
state(176 downto 93) <= state(175 downto 93) & t1;
state(287 downto 177) <= state(286 downto 177) & t2;
```

After that, the key stream generation consists mainly of an iterative process which updates some bits in the state register with logic operations to generate one bit of key stream. The VHDL code description is given by the following lines:

```
k1 <= state(65) XOR state(92);
k2 <= state(161) XOR state(176);
k3 <= state(242) XOR state(287);

a1 <= state(90) AND state(91);
a2 <= state(174) AND state(175);
a3 <= state(285) AND state(286);

t1 <= k1 XOR a1 XOR state(170);
t2 <= k2 XOR a2 XOR state(263);
t3 <= k3 XOR a3 XOR state(68);

keystream <= k1 XOR k2 XOR k3;
```

The schematic representation of the Trivium radix-1 algorithm (which outputs one key stream bit in every clock cycle) consists of three circular shift registers of different lengths and any combinational logic to perform the addition (exclusive-or cells) and multiplication (and cells) operations as the schematic representation version shown in fig.1. The length of each shift register is different; the first register has 93 bits, the second 83 bits and the third 111 bits. The key stream is the result of exclusive-or operations on some bits in the shift register [9].

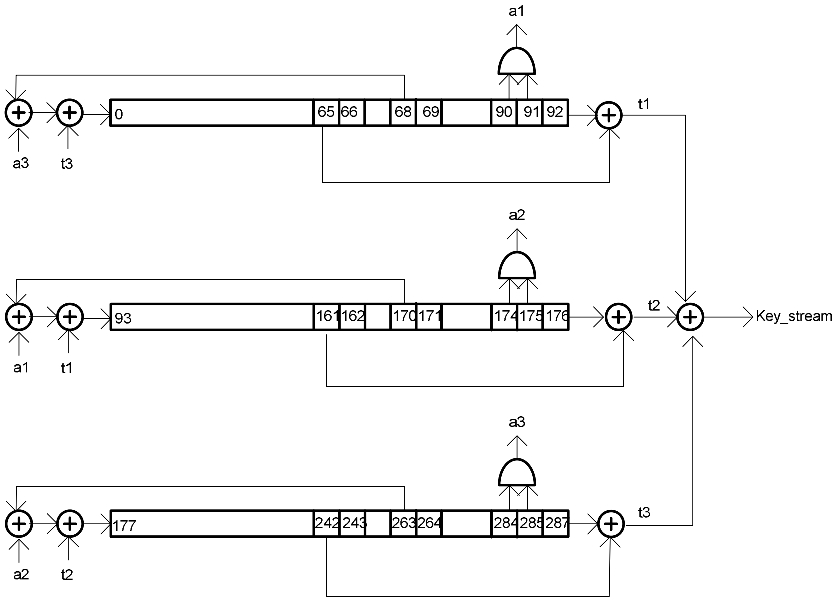


Fig. 1. Trivium schematic representation

The main cost (area and power) of this implementation is in the registers. Trivium radix-1 is a good choice for evaluating the validity of the power reduction technique. Other radix versions allow fast, power-efficient implementations with the same number of storage elements but involve more combinational logic and more complex designs [4].

3 Power Reduction in Trivium by Logic Parallelization

The Trivium low power implementation presented here is based on shift register logic parallelization [5], [6], [7]. The main idea of this technique is to divide a shift register in two, each half-length being like the original. During shift register operation, the data coming in even cycles is stored in one register and that arriving in odd cycles is stored in the other. One of the registers is called the "even register" and the other the "odd register". In this way, the clock which triggers the loading of each of these registers has a frequency which is half the original frequency.

Figure 2 graphically shows the structure of the register division. The odd group is synchronized with the clock rise time, while the even group is synchronized with the clock fall time. In every clock cycle each shift register stores a new bit and places another bit in its most significant position. Of the most significant bits supplied by the registers even and odd bit must be selected that provides the output. To do so, a multiplexer is used, controlled by the clock signal of the registers.

Power reduction is achieved because in each clock cycle only half of the data is shifted. Thus, each piece of data has to go through half of the number of flip-flops to reach the output. However, it is necessary to introduce additional circuitry to divide the clock, and a multiplexer to select the data in the output. This technique is therefore especially effective in reducing power consumption in medium-sized or large shift registers.

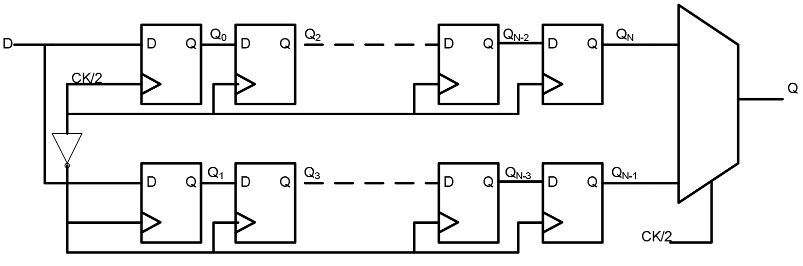


Fig. 2. Parallel Shift Register

Some modifications are required for the application of this technique to the Trivium implementation. Firstly, as shown in Fig.1, the Trivium implementation has three different circular shift registers. With the parallelization technique these are separated into six shift registers, as is shown in Fig 3. Each shift register is divided into an odd and even shift register with half bits. The length of each shift register is indicated in the figure inside the register.

Secondly, generation both of the input bits in each of the shift registers and the output bits depends on the bits stored in different positions in the shift registers. These positions, depending in turn on the clock cycle, will match a bit set in the even or in the odd register. It is therefore necessary to introduce a logic that allows these bits to be selected correctly.

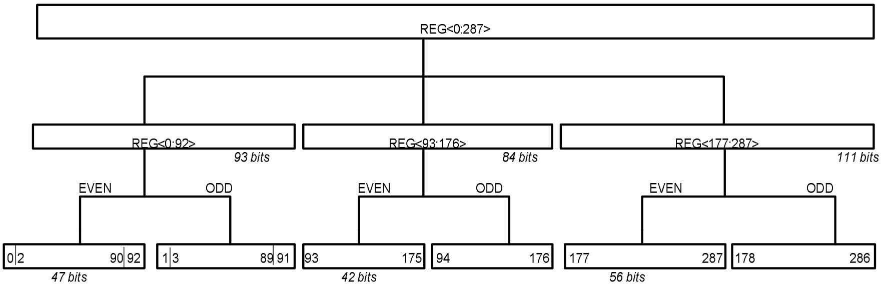


Fig. 3. Even and odd shift registers

As shown in Fig. 4, this logic basically means introducing multiplexers which, using the clock signal as the selection signal, will select the bit to be taken from the shift register.

Thirdly, in the first clock cycles both the key and initialization vector must be loaded in parallel. In this implementation, the load must be maintained in parallel, but taking into account that the even registers are loaded with the rising edge and the odd registers are loaded with the falling edge of a clock with half the frequency of the original clock. To solve this loading problem we decided to use two clock cycles, with the even registers being loaded in the first cycle and the odd registers being loaded in the second.

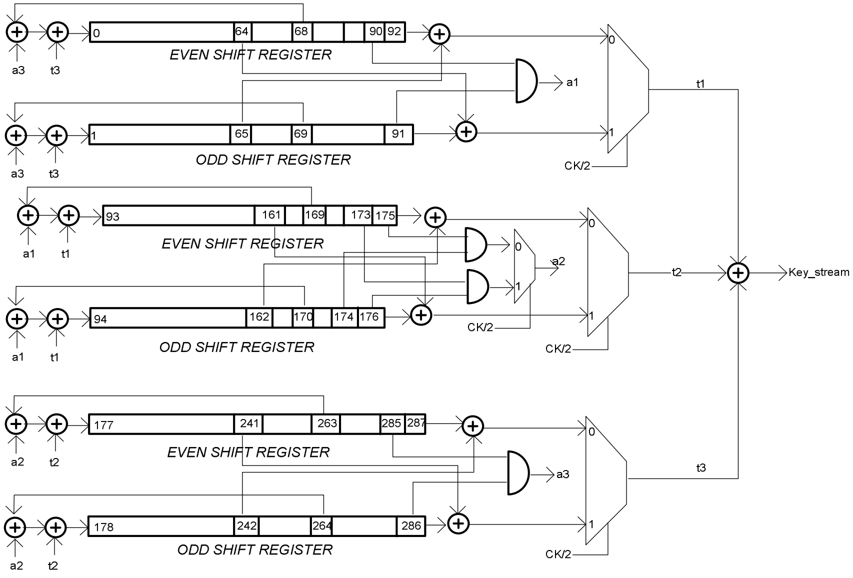


Fig. 4. Schematic of Low Power Trivium implementation

Other low power Trivium implementations were described beforehand. The implementation described in [8] uses a clock gating technique and a radix-16 datapath. Although the clock gating technique is widely used, it is not very useful when applied to shift registers. In fact, in [8] no data comparisons between low power and standard implementations are shown.

3.1 Trivium Implementations

The low-power and standard versions of Trivium were described and designed using VHSIC Hardware Description Language (VHDL). The resulting implementations were verified using the Mentor Graphics ModelSim simulation environment, with test vectors using a key presented in the reference files of Trivium [2]. The technologies used were CMOS 180 nm, 130 nm and 90 nm. The standard version of Trivium was also implemented, so the results obtained by both implementations can be compared.

The timing and area reports for the low power (LP) and standard (TRIV) versions provided by the Design Vision synthesis tool are shown in Table 1. The timing and area data depends on the standard cell library of each technology. Our aim is not to compare the results of the all three technologies, but to check that a significant reduction in power consumption is achieved in them.

The low power version uses more standard cells (4-7%) than the standard version, although the number of cells decreases when the technology is smaller. This happens because the synthesis tool chooses the cells from the available ones in the library, and the libraries are different in the three technologies. The non-combinational logic area must be quite similar in both designs, because the number of flip-flops do not change (only the flip-flops for the clock division are introduced). But in 130 and 90 nm the synthesizer uses different flip-flops, with less area in the low power version than in the standard version, so the non-combinational area is similar or less. In conclusion, when the technology decreases the gate size, the cell count rises slightly by 6.6% in 180 nm and by 3.5% in 90 nm in the low power version and the cell area penalty is not as huge as might be expected since the cell area increases by 6.4% in 180 nm and by 2.2% in 130 nm but decreases by 0.6% when the technology is 90 nm because it uses flip-flops with less area from the technology library.

With respect to delay, table 1 shows that both implementations achieve the 40 ns restriction imposed on the clock. The multiplexer delay slows the low power implementation down slightly in comparison with the standard implementation.

Table 1. Synthesis Report

Report		*180nm*		*130nm*		*90nm*	
		TRIV	*LP*	*TRIV*	*LP*	*TRIV*	*LP*
Timing(ns)	*Critical Path Length*	2.72	1.08	2.20	0.34	1.35	0.18
	Critical Path Slack	37.19	38.86	37.60	39.50	38.58	39.73
	Critical Path Clk Period	40.00					
Cell Count		604	644	608	637	602	623
Area(μm^2)	*Combinational Area*	8265	8634	2724	3401	1747	2191
	Noncombinational Area	14694	15795	9677	9281	5644	5154
	Cell Area	22959	24429	12401	12682	7392	7345

3.2 Power Consumption

Power consumption in the design was calculated with Synopsys tools and a switching activity file in SAIF (switching activity interchange format) format, generated with simulations in Modelsim with typical case timing analysis for a desired clock rate of 20 MHz. Power modelling was performed using the typical foundry values for each process.

Power dissipated can be divided in the Synopsys´s report into cell leakage (static) and dynamic power. Static power is the power consumed by a gate when it is not switching. It is caused by currents flowing through the transistors when they are turned off and it mainly depends on the size of the circuit. Dynamic power is the power dissipated when the circuit is active. Dynamic power is further divided into net switching power and cell internal power.

Net switching power is the power dissipated by net interconnects and gate capacitance. The amount of net switching power depends on the switching activity (and therefore related to the operating frequency) of the cell. The more logic transitions in the cell output, the higher the switching power. Lowering the circuit size and reducing the supply voltage also directly reduces dynamic power. Cell internal power is consumed within a cell by charging and discharging internal cell capacitances. Internal power also includes short-circuit power. In these technologies static power is much lower than dynamic power.

When comparing the power consumption of the two implementations, shown in Table 2, we noticed that the low power version always has lower cell internal power consumption than the standard version: 41% lower in 180 nm, 33% in 130 nm and 25% in 90 nm. Cell internal power consumption therefore decreases with smaller technologies. However, net switching power increases in the low power version (16%, 77% and 70%) compared to the standard version due to the rise in the number of net connections. These different switching power values increase in the three technologies due to the cells chosen by the synthesis tool for the combinational part.

The most useful comparison was made when the total dynamic power of the standard and the low power Trivium designs was evaluated. The results show that dynamic power consumption in the low power version is lower, in all technologies, than dynamic power consumption in the standard version. In 90nm it decreased by a factor of about 18%; in 130 nm, by a factor of 23% and in 180 nm by 29%.

Table 2. Power Report

Power@20MHz	*180nm*		*130nm*		*90nm*	
	TRIV	*LP*	*TRIV*	*LP*	*TRIV*	*LP*
	Core Voltage = 1.8V		*Core Voltage = 1.2 V*		*Core Voltage = 1.2 V*	
Cell Internal (μW)	868	550	218	145	203	152
Net Switching (μW)	139	162	18	33	16	27
Total Dynamic (μW)	1007	712	236	178	219	179
Cell leakage (nW)	89.9	102.9	249.7	290	17.6	19.27

4 Summary and Conclusions

In this paper we have described and compared a new low power Trivium implementation based on logic parallelization techniques. We have synthesized the implementation in three technologies with different gate lengths - 180nm, 130nm and 90 nm - to compare area penalty and power consumption results.

The area's penalty and cell number obtained with this technique is very low (less than 6%) while the improvement in dynamic power consumption is quite high (more than 18%). For instance, in 180 nm a reduction of 29% was obtainable in dynamic power with an area penalty of only about 6.4%. The best solution was in 90 nm, where dynamic power was reduced by a factor of 18% and cell area decreased slightly by a factor of 0.6%.

We think this solution might be used in designs where the slight increase in cell number requirements is less important than power reduction. In lower technologies it could be used for passively-powered applications like RFID tags, smart cards and Bluetooth products.

Acknowledgments. This work was partially funded by the following projects: Cripto-Bio, Junta de Andalucía (Andalusian Regional Government) (P08-TIC-03674); MOBY-DIC, EEC (FP7-ICT-2009-4) and CITIES, MCINN (TEC2010-16870/MIC).

References

1. Diffie, W.Y., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory 22, 644–654 (1976)
2. eSTREAM: ECRYPT Stream Cipher Project, `http://www.ecrypt.eu.org/stream/`
3. Ver., `http://www.ecrypt.eu.org/stream/D.SYM.3-v1.1.pdf`
4. De Canniere, C., Preneel, Y.B.: Trivium, A Stream Cipher Construction Inspired by Block Cipher Design Principles, eSTREAM, ECRYPT Stream Cipher Project, `http://www.ecrypt.eu.org/stream/papersdir/2006/021.pdf`
5. Piguet, C., et al.: Logic Design for Low-Voltage / Low- Power CMOS Circuits. In: Proc. of the 1995 International Symposium on Low Power Design, Dana Point, CA (April 1995)
6. Schneider, T., et al.: Low-Voltage Low-Power Parallelized Logic Modules. In: Proc. PATMOS 1995, Paper S4.2, Oldenburg, Germany, October 4-6 (1995)
7. Piquet, C.: Low-Power CMOS Circuits technology, Logic Design and CAD Tools. CRC Press (2006)
8. Feldhofer, M.: Comparison of Low-Power Implementations of Trivium and Grain. eSTREAM, ECRYPT Stream Cipher Project, `http://www.ecrypt.eu.org/stream/papersdir/2007/027.pdf`
9. Schaumont, P.R.: A Practical Introduction to Hardware/Software Codesign. Springer Science+Business Media, LLC (2010)
10. Robshaw, M., Billet, O. (eds.): New Stream Cipher Designs. The eSTREAM Finalists. Springer (2008)

A Generic Architecture for Robust Asynchronous Communication Links

Jakob Lechner and Robert Najvirt

Institute of Computer Engineering, Vienna University of Technology, Austria
{lechner,rnajvirt}@ecs.tuwien.ac.at

Abstract. This paper proposes a new generic architecture for building robust communication links for globally asynchronous locally synchronous (GALS) circuits. The general idea is to use delay-insensitive codes along with error detecting codes to provide resilience against transient faults as well as robustness against delay variations. The presented link architecture is completely generic with respect to the chosen handshake protocols (2-phase/4-phase) and the used codes. Thus a specific implementation can be individually optimized regarding features such as performance, power consumption, area complexity or the number of faults that can be tolerated. In order to demonstrate the flexibility of our approach we present several solutions based on 2-phase and 4-phase dual-rail codes combined with either single parity bits or Hamming codes for error detection. In the former case the link provides resilience against single faults, in the latter double faults can be mitigated.

1 Introduction

The trend of past and still ongoing aggressive technology scaling for better performance and power consumption is associated with many challenging design problems. Although it allows for integration of complex systems-on-chip (SoC) with billions of transistors on a single die, clocking the whole chip with a single, high-frequency clock has become infeasible. As a possible solution, globally asynchronous locally synchronous (GALS) systems consist of multiple small, locally clocked, synchronous modules communicating over asynchronous links. A global clock net and the associated timing closure problems are thus avoided. On the other hand, timing problems are also caused by increasing process, voltage and temperature (PVT) variations. This requires large timing margins for acceptable error rates. Furthermore reliable circuit operation is threatened by smaller feature sizes and smaller critical charges, which make transistors increasingly vulnerable to radiation faults, electromagnetic interference or noise.

In this paper[1] we present a new architecture for asynchronous communication links as needed in GALS systems. The aforementioned reliability issues are tackled using delay-insensitive codes in combination with error detecting codes.

[1] This work is supported by the Austrian Science Foundation (FWF): FATAL project (P21694) and PSRTS project (P20529-N13).

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 121–130, 2013.

While the former allow for adaption to varying propagation delays and thus prevent timing faults, the latter provide fault-tolerance in the value domain. A big advantage of the proposed solution is the fact that it can be used with any delay-insensitive and error detecting codes.

This paper is organized as follows: Section 2 gives a brief introduction to asynchronous communication and discloses the assumed fault model. In Section 3, the generic architecture of the proposed communication link is introduced and discussed. In the following section a specific implementation using 2-phase and 4-phase dual-rail codes is presented. Section 5 then shows an evaluation of these dual-rail links and presents performance and area results. Finally, Section 6 puts our solution in context with existing work on fault-tolerant asynchronous communication links. Section 7 concludes the paper and outlines future work.

2 Preliminaries

2.1 Asynchronous Communication

There is a great variety of asynchronous communication protocols. In general, the used protocols are handshake-based, i.e., they have some form of a request and acknowledge signal. If every transition of the request signal indicates new data, this is called a 2-phase protocol. Alternatively, in 4-phase protocols new data is only delivered upon a rising edge of the request signal. Thus, every communication cycle is ended by a reset of the handshake signals.

Furthermore asynchronous communication channels can be characterized by the way the request signal is delivered. It can either be sent on a separate control wire or it can be encoded along with the transmitted data word. In the first case, the timing of the request wire needs to be *matched* with the *bundled* data signals. This approach therefore is called *matched delay* or *bundled data*. In the latter case, special codes are used that allow for completion detection: The receiver then is able to tell whether the transmission is complete by simply waiting for the arrival of a valid code word. This is called *delay-insensitive* (DI) communication as no timing assumptions have to be made.

2.2 Fault Model

All assertions about fault-tolerance made in this paper rely on the following assumptions about faults occurring in the system: a) All faults are of transient nature, b) faults only occur in the interconnect, c) no more than f faults occur during one communication cycle, f being the maximal number of errors the used error detecting code can reliably detect, d) the length of metastable upsets is bounded and known.

Assumption a) makes it possible to resolve received errors by waiting for the transient faults to cease and thus avoiding the need for complex error handling schemes such as forward error correction or retransmissions. Transient faults are erroneous transitions changing the state of the wire in either of the two ways: $0 \rightarrow 1 \rightarrow 0$ or $1 \rightarrow 0 \rightarrow 1$. Implicitly, assuming transient faults requires disallowing the interconnect to contain state holding elements, i.e., not to be pipelined. This is because faults could cause upsets in unprotected state holding elements of the interconnect leading to soft errors persistent for the duration of the communication cycle. Assumption b) is used for simplifying the discussions in this paper. In a fully fault-tolerant system, the sender and receiver components would of course be protected by some kind of fault-tolerance mechanism, such as triple modular redundancy. Assumption c) naturally restricts the discussion to cases where the error detection mechanism will not fail. The last assumption is concerned with metastability, which can occur when faults violate the setup/hold window of flip-flops while input data is latched. Choosing a reasonable upper bound allows for building circuits that will only fail with a very low probability.

3 Link Architecture

The proposed link architecture basically consists of input and output ports that provide an asynchronous communication link between two synchronous GALS modules. Clearly, a module that sends data needs to be equipped with an output port, whereas the receiver module is hooked up with an input port. Many GALS systems use stoppable clocks to avoid synchronization problems during asynchronous I/O operations. Since the ports obviously need to remain operational when communicating, they have to be implemented as self-timed circuits. The local data exchange between a GALS module and its associated I/O ports is performed using a simple bundled data protocol.

Input and output ports are assumed to be connected over a global on-chip interconnect that is susceptible to transient faults and where signal delays can vary with the operating conditions (PVT variations). To allow for reliable communication we propose to use a combination of *delay-insensitive* (DI) and *error detecting* (ED) codes. The basic idea is to verify both *completeness* and *correctness* of a transmitted data word at the receiver side. If faults occur, the input port will wait for them to disappear and will only acknowledge the reception once the correct data word has been latched. In the following, the structure of the output and input port will be described in detail. Note that the presented architecture is completely generic with respect to the type of handshake protocol (2-phase/4-phase) and the used delay-insensitive and error detecting codes.

3.1 Output Port

The structure of the output port is rather simple and directly follows from the basic concept described above. As can be seen in Fig. 1, the ED encoder first augments a new data word arriving from the sender with the required redundancy to enable error detection. Subsequently, the produced data word is encoded with a DI code, resulting in a code word ready to be transmitted. The control logic then is responsible for clocking the output register in order to put this code word on the interconnect signals. Two conditions need to be satisfied

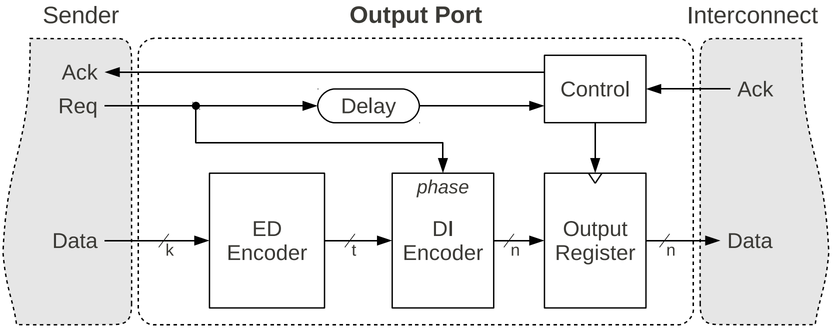


Fig. 1. Output Port

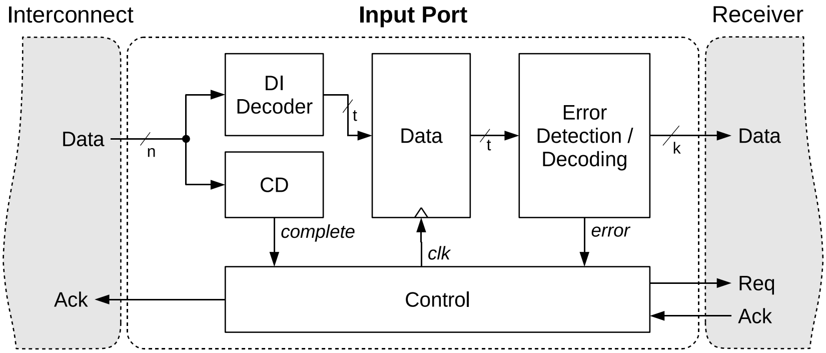


Fig. 2. Input Port

before a clock pulse is produced: a) New data needs to be ready as indicated by the request signal from the sender, b) the last transmitted code word needs to be acknowledged by the input port of the receiver. Note that the delay element on the request signal needs to be matched to the maximal propagation time of the data path, i.e., the ED and DI encoders. This ensures that a new code word safely stabilizes at the register input before the clock pulse is generated.

3.2 Input Port

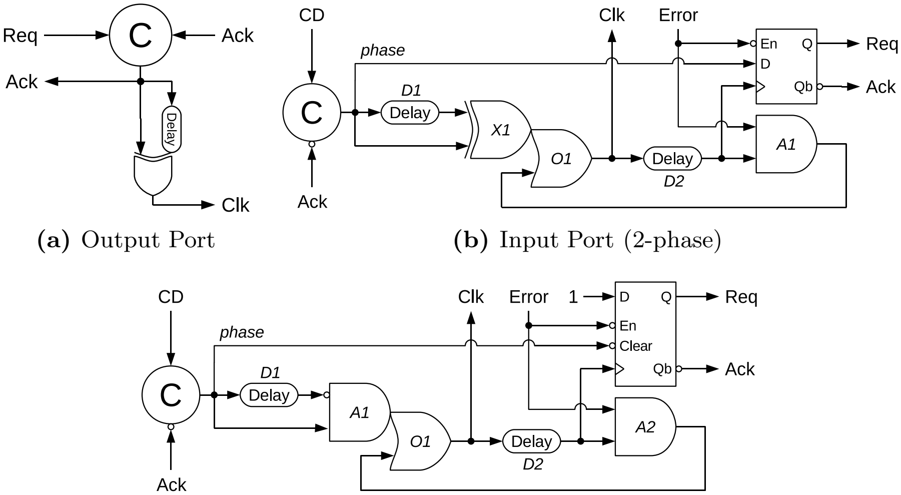
In the input port an incoming code word is first checked for completeness. This is performed by a completion detector (CD), as can be seen in Fig. 2. In case of a 2-phase protocol every complete data word that arrives will toggle the output of the CD unit between one and zero. In a 4-phase protocol the output of the completion detector also changes between one and zero, but only rising transitions indicate the availability of new data. Recall that a handshake cycle in a 4-phase protocol is ended with a reset phase. This is typically done by resetting all interconnect wires to zero. Such an all-zero input vector is called *spacer code word*, as it separates successive data words.

Once a new data word is available, the control unit, connected to the output of the completion detector, will issue a clock pulse, which latches the data word into the input register. Note that the DI decoder is placed before the register. This allows for concurrent operation with the completion detection and also reduces the size of the input register. After the new data word has been stored, error detection can be performed. If no error is detected, the correct data word can be forwarded to the receiver: The control unit waits until error detection is finished and then checks the *error* output of the ED decoder (see Fig. 2). If this signal is zero, a request transition is produced for the receiver and an acknowledge is returned back to the sender's output port.

In case an error is detected, the control unit simply issues another clock pulse. Hence, the data word is sampled again and analyzed by the ED unit. This is repeated until the input register contains a fault-free data word. Note that this will eventually happen since we only assumed transient faults in our fault model. Of course, it would also be possible to sample the input data just once and use forward error correction to recover faulty bits. However, stronger codes would be needed that add more check bits and therefore increase the number of required interconnect wires. Furthermore this approach would increase the size of the input port and reduce the performance of fault-free transmissions.

3.3 Control Circuits

In this subsection we want to present the control circuits we have implemented for the proposed link architecture. Note that these circuits are as well completely independent from the employed delay-insensitive and error detecting codes. The controller for the output port is depicted in Fig. 3a. It can be used both for



(a) Output Port

(b) Input Port (2-phase)

(c) Input Port (4-phase)

Fig. 3. Control Circuits

4-phase and 2-phase protocols. A C-gate simply waits for a new request from the sender module and an acknowledge from the receiver. If both signals are equal, new data can be transmitted. The output of the C-gate then makes a transition, either falling or rising – depending on the protocol phase. This transition forks into two wire branches that are connected to the same 2-input XOR-gate. As one branch is delayed, the inputs of the XOR-gate will be unequal for a short time and a $0 \rightarrow 1 \rightarrow 0$ pulse is produced at the output of the XOR-gate. This pulse clocks the data register of the output port and a new data word/spacer word is issued. Concurrently the output transition of the C-gate is returned to the sender as acknowledge signal.

The implementation of the input port controllers is a little more intricate. This time, different circuits are needed for 2-phase and 4-phase protocols. Let us first consider the 2-phase version (Fig. 3b). The left-hand side of the circuit is the same as the output port controller: A C-gate waits for a transition on the signal from the completion detector and forwards it if the receiver is ready (see ack signal connected to the lower C-gate input). Due to XOR-gate *X1* an up/down pulse is produced, which is then propagated to the clock output. As explained in the previous section, this clock pulse triggers the sampling of the input data, and thereby initiates error detection. If no error was found, the transition of the completion detection needs to be forwarded to the request output to notify the receiver. Furthermore the acknowledge signal needs to be toggled to inform the sender about successful data reception. This task is performed by a flip-flop on the right-hand side of the circuit. Note that this flip-flop is only enabled if the error signal is zero and is clocked by a delayed version of the initially generated clock pulse (see delay *D2* in Fig. 3b). Obviously, the rising clock transition needs to be delayed long enough so that the error detection has completed and the error/enable signal is stable. Thus, delay element *D2* needs to be matched to the data path of the input port (i.e., clock-to-output delay of the input register plus propagation time of the error detector plus setup-time of the flip-flop).

In case a fault was detected ($error = 1$), the flip-flop will not change the request/acknowledge signals. Instead the delayed clock pulse is re-used to issue another clock tick for sampling the input data once again. Since the error signal is high, a feedback loop is established by AND-gate *A2*. The clock pulse therefore can propagate on the feedback signal and appears on the clock output again. Clock pulses will be generated this way until the transient fault disappears. In theory this could go on for an infinite number of cycles. In practice, however, the oscillation will eventually decay due to pulse broadening or shortening. Nevertheless, if the pulse width is adjusted properly (controlled by delay *D1*), the transient error will vanish long before the oscillation stops.

The 4-phase controller, which can be seen in Fig. 3c, looks very similar. However, there are some important modifications. In a 4-phase protocol a clock pulse only needs to be generated for the data phase. The spacer code word, transmitted during the reset phase, just serves for resetting the interconnect signals. It is not necessary to sample it or forward it to the receiver. Therefore, a clock pulse is only generated for *rising* output transitions of the C-gate. As before, the output of the C-gate forks into two branches, but this time connected to an AND-gate (*A1*) instead of an XOR-gate. Note that the delayed branch is inverted. It can be easily seen that rising transitions will produce a pulse on the AND-gate output, whereas falling transitions will be masked.

However, a falling transition at the C-gate's output during the reset phase has another effect. As can be seen in Fig. 3c, the output of the C-gate is connected with active-low clear input of the flip-flop, which controls the request/acknowledge signals. Thus, the flip-flop will be immediately reset to zero, which completes the current handshake cycle. Since there is no need to sample input data or perform error detection, the reset phase of the communication cycle is significantly shorter than the data phase.

4 Dual-Rail Implementations

For evaluation of the link architecture presented in the previous section, we decided to implement input and output ports that use dual-rail codes for delay-insensitive communication across the interconnect. Dual-rail codes are often used in asynchronous circuit designs and, due to their simplicity, can be implemented with small and fast encoder/decoder units. As the name suggests, these kind of codes transmit every data bit over two wires. This allows for data encodings that provide the desired delay-insensitivity. In case of a 4-phase protocol a 1-of-2 code is used, for a 2-phase protocol the encoding is called LEDR (level-encoded dual-rail). Detailed information on these codes and the implementation of circuits for encoding, decoding and completion detection can be found in [1].

Concerning the ED codes we have implemented ports that protect data word with a single parity bit or alternatively with Hamming codes. A single parity bit increases the minimum hamming distance d between protected data words to 2. Therefore, $d - 1 = 1$ bit faults can be detected. Hamming codes have a distance of 3, thus allowing for detection of up to 2 bit faults. Depending on the data width, a Hamming code adds a certain number of parity bits [2].

For assessing the overall resilience of a specific implementation, the used delay-insensitive code has to be considered as well. It needs to be investigated how faults in the DI code word relate to faults in the decoded data word. In general, the number of faults introduced in a delay-insensitive code word does not necessarily need to be equal to the number of resulting bit faults in the decoded data word. In case of dual-rail codes, fortunately, things are simple. As every dual-rail pair encodes exactly one data bit, f faults in the dual-rail code word can cause at most f faults in the decoded data word. Thus, a single parity bit enables resilience against *single faults* on the interconnect signals, whereas *double faults* can be mitigated with Hamming codes.

5 Results

To be able to test and evaluate the proposed dual-rail links we have implemented the following four specific architectures:

- 2-phase dual-rail (LEDR encoding) with single parity bit
- 2-phase dual-rail (LEDR encoding) with Hamming code
- 4-phase dual-rail (1-of-2 code) with single parity bit
- 4-phase dual-rail (1-of-2 code) with Hamming code

Based on VHDL descriptions of these links, we performed logic synthesis and mapped the circuits to a UMC 90nm standard cell library. A timing simulation of the synthesized netlist for 2-phase ports with Hamming code protection can be seen in Fig. 4. The simulation waveform shows two transmissions. For the first transfer no faults interfere during the communication. A single clock pulse is generated and the incoming data word is forwarded to the receiver. Note that the short pulse on the error signal is just a glitch that occurs while the error detection circuit performs its computations. During the second transmission we have injected a double fault on the interconnect signal (two transitions in the red circle marked with the flash symbol). As can be seen, the error signal is raised after the input data are latched for the first time. Thus, no request is produced for the sender. Instead, a second clock pulse is generated. Since the faults have vanished at this point in time, the error signal is reset to zero and the transmission can be completed.

Table 1 shows area and performance characteristics. As can be seen, we have benchmarked different data widths for the first two port designs. Regarding the area complexity the table nicely shows that the circuits scale linearly with increasing number of data bits. The coding efficiency, i.e., the ratio of transmitted data bits to required interconnect wires, is shown in the last column. Since we are using dual-rail codes, the efficiency can of course never be above 0.5.

The performance results are derived from simulations, where we assumed ideal conditions to attain the maximum throughput and minimum latencies. Thus, we did not introduce additional delays on the interconnect signals and used simulation models for ideal sender and receiver components that produce and consume handshake requests in zero time. The cycle time represents the duration of a complete handshake cycle (including the reset phase in case of a 4-phase protocol). Latency values show the time it takes for a request transition of the sender to propagate across the I/O ports to the receiver. We have measured latencies both for fault-free and faulty transmissions. In the latter case we only

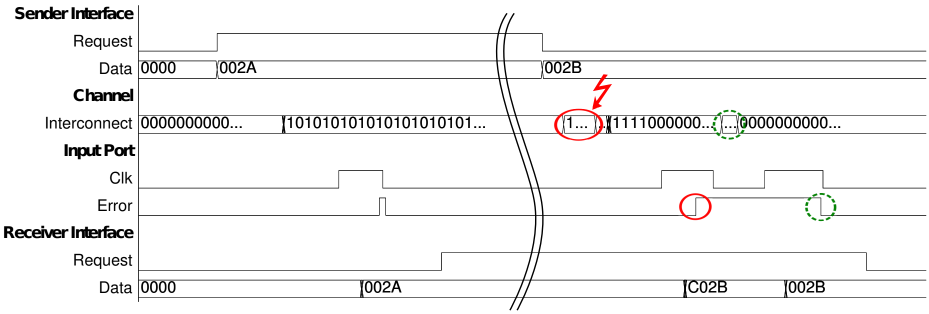


Fig. 4. Simulation: Two transmissions, with and without faults

Table 1. Performance & Area Evaluation

Protocol/ED Code	Data Width	Cycle	Throughput	Latency (ns)		Area	Coding
	(bit)	(ns)	(MHz)	no fault	fault	(μm^2)	Efficiency
2-phase/Single Parity	16	1.17	855	1.27	1.78	1767	0.47
	32	1.27	789	1.43	2.02	3278	0.48
	64	1.38	723	1.57	2.23	6523	0.49
4-phase/Single Parity	16	1.79	559	1.37	1.92	1642	0.47
	32	1.91	524	1.50	2.11	3132	0.48
	64	2.14	467	1.73	2.41	6206	0.49
2-phase/Hamming	16	1.31	766	1.42	2.08	2340	0.38
4-phase/Hamming	16	1.95	513	1.54	2.21	2262	0.38

injected faults that vanish before the second sample is taken by the receiver. As can be seen in Table 1, the 2-phase implementations significantly outperform their 4-phase counterparts regarding throughput and latency. This is possible because of the non-existing reset phase. Hence, 2-phase protocols clearly are the better choice for asynchronous communication channels.

6 Related Work

In an earlier publication [3] we have already presented a circuit architecture for building robust asynchronous communication channels. A 4-phase dual-rail code in companion with a single parity bit was used as well. The method for recovering incorrect data words, however, is completely different. Instead of taking multiple input samples, the input port latches a complete input word only once. Single bit faults can then be corrected with the combined redundancy of the dual-rail code and the parity bit.

The idea of using error correcting codes in combination with delay-insensitive codes has been proposed before in [4]. This paper presents different fault models for delay-insensitive data transmission and gives theoretical results on the necessary error correction capabilities. The paper also presents a short draft based on a parity bit-protected 4-phase dual-rail code, like we use in our approach. However, no implementation details or circuit designs are disclosed.

[5] proposes fault-tolerant asynchronous links for NoC systems. Data bits are transmitted over dual-rail signals, the used encoding, however, is completely different from classic dual-rail codes like 1-of-2 or LEDR. Unfortunately, a close examination of the concept reveals that it does not introduce more fault-tolerance than the temporal masking in the completion detection when an additional (random) dual-rail pair is transmitted.

In [6,7] a new family of error correcting/delay-insensitive codes, called ZeroSum codes, is introduced. These codes can correct all single-bit errors and depending on the specific type of ZeroSum code also a certain number of double-bit errors. However, error correction and delay-insensitivity cannot be provided at the same time, i.e., faults can transform incomplete intermediate input vectors into complete code words, which have to be accepted by the receiver.

7 Conclusion and Future Work

In this paper, we have presented a novel, generic architecture for robust asynchronous communication links targeting GALS systems. The proposed links provide tolerance of variable interconnect delays by using delay-insensitive codes as well as protection against transient faults through error detecting codes. While giving a detailed gate-level implementation for the control logic, any standard implementation for encoders/decoders and both 2-phase and 4-phase handshake protocols can be chosen, giving designers great options for optimization. In this design space we then presented four possible implementations, all based on dual-rail codes, and analyzed their performance and area characteristics.

Concerning future work we plan to investigate how the port designs can be adapted to make them metastability-tolerant. In [3] we have proposed a specific solution for 4-phase dual-rail codes with a special input port that waits for metastable upsets to resolve. Unfortunately, a more general approach that can also be applied to 2-phase protocols is much more complicated. Another plan for future work is to investigate other delay-insensitive codes to improve the coding efficiency and therefore decrease the number of required interconnect signals.

References

1. Sparsø, J., Furber, S. (eds.): Principles of Asynchronous Circuit Design: A Systems Perspective. Kluwer Academic Publishers (2001)
2. Koren, I., Krishna, C.M.: Fault Tolerant Systems. Morgan Kaufmann Publishers Inc., San Francisco (2007)
3. Lechner, J., Lampacher, M., Polzer, T.: A robust asynchronous interfacing scheme with four-phase dual-rail coding. In: Seventh International Conference on Application of Concurrency to System Design, ACSD 2012 (June 2012)
4. Cheng, F.C., Ho, S.L.: Efficient systematic error-correcting codes for semi-delay-insensitive data transmission. In: Proceedings of the 2001 International Conference on Computer Design, ICCD 2001, pp. 24–29 (2001)
5. Ogg, S., Al-Hashimi, B., Yakovlev, A.: Asynchronous transient resilient links for noc. In: Proceedings of the 6th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS 2008, pp. 209–214. ACM, New York (2008)
6. Agyekum, M., Nowick, S.: An error-correcting unordered code and hardware support for robust asynchronous global communication. In: Design, Automation Test in Europe Conference Exhibition (DATE), pp. 765–770 (March 2010)
7. Agyekum, M., Nowick, S.: Error-correcting unordered codes and hardware support for robust asynchronous global communication. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 31(1), 75–88 (2012)

Direct Statistical Simulation of Timing Properties in Sequential Circuits

Javier Rodríguez, Qin Tang, Amir Zjajo,
Michel Berkelaar, and Nick van der Meijs

Circuits and Systems Group, Delft University of Technology,
Mekelweg 4, 2628 CD Delft, The Netherlands
J.RodriguezRodriguezdeGuzman@student.tudelft.nl

Abstract. Accurate timing analysis of digital integrated circuits is becoming harder to achieve with current and future CMOS technologies. The shrinking feature sizes lead to increasingly important local process variations (PV), making existing methods like corner-based static timing analysis (STA) yield overly pessimistic results. In this paper we propose a general purpose statistical circuit simulator for accurate timing analysis. A statistical simplified transistor model (SSTM) is used as the simulator's building block, allowing accurate simulation of sequential circuits while fast statistical analysis is achieved by solving a system of random differential equations (RDE), thus avoiding time-consuming Monte Carlo simulations. The conducted experiments show the accurate calculation of crossing time statistical moments for several sequential cells using 45 nm CMOS technology.

1 Introduction

CMOS technology nodes below 45 nm are currently the state of the art in the semiconductor industry. As the transistor feature sizes are continuously being reduced, process variations (PV, typically random deviations from the intended nominal values) have an increasingly significant impact on chip performance and cannot be ignored by the design and verification tools. This fact is specially critical when dealing with synchronous designs which have to meet strict timing constraints.

Circuit transient simulation is the most accurate way to check a circuit's timing compliance (also known as Dynamic Timing Analysis, DTA). However, DTA's exponential time complexity has made Static Timing Analysis (STA) [1,2], which is a linear time complexity technique, the standard timing verification tool for the last 20 years. STA is typically used along with corner analysis, which calculates a best/worst case scenario for each parameter that may affect the circuit's performance, thus providing conservative bounds for each circuit delay. In this sense, this method can be seen as an *inter-die* PV set up, i.e. all the transistors on the same die are supposed to be affected in the same way. However, if this assumption is not true, as it is in the case of local *within-die* PV, different correlations between path delays will cause highly pessimistic estimates

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 131–141, 2013.

or even optimistic estimates, depending on the circuit's topology [3]. Besides, as transistor features shrink with each technology node, the number of parameters and hence corners to take into account are too big to keep these kinds of methods attractive in terms of runtime.

Statistical STA (SSTA) was created to deal with STA's shortcomings by modeling the gate delay as probability distributions instead of deterministic data, which can handle adequately spatial correlations. Accurate timing estimates are expressed now, in terms of timing yield, i.e. the probability of a circuit to meet timing constraints. However, while STA algorithms require deterministic *sum* and *max/min* operations, their statistical counterparts are, in general, not trivial, specially in the presence of spatial correlations [4]. Some approximations to this problem assume normal distributions expressed in a first order canonical form. Then, *sum* operation becomes trivial while *max/min* is mainly approximated by forcing its output to be also a normal distribution in canonical form, whose coefficients are computed using a probabilistic based weighting [5] or a linear time upper bound [6].

Timing analysis tools use mainly gate-level delay models for standard cells characterization since they are the basic building blocks for most circuits and simplify their analysis. Simple LUT-based models, like the Non-Linear Delay Model (NDLM), assume a saturated input voltage ramp and capacitive load. However these assumptions do not hold for modern circuits with increasing crosstalk noise and complex resistive wire interconnections, therefore making NLDMs not an accurate timing model any more. In [7], a noise aware Current Source Model (CSM) is presented for combinational cells using a voltage controlled current source, modeled as a 2D-LUT with input and output voltages as the table indices, and a linear output capacitance. Transient simulation with this model allows accurate timing analysis for arbitrary input signals and loads. This model was extended in [8] to cope with the increasing importance of device parasitics and, later in [9], to provide statistical timing analysis by extended Monte Carlo generated LUTs to characterize the CSM's PV sensitivities.

Up to this point, all the proposed models have been focused on combinational logic cells, but none or very few of them deal with sequential cells, although these elements constitute an essential part of timing analysis, thus requiring a really accurate characterization. Clock to output delay ($\mathrm{T}_{\text{CLK-O}}$) is computed by typical STA tools using simple NLDMs, under the assumption of stable input signals before and after the clock edge (e.g. setup and hold times). Some efforts have been done to improve this method's accuracy, like exploiting setup and hold times interdependence [10], but this approach still leads to pessimistic and inaccurate timing estimates, specially if within-die PV are present.

In the presence of arbitrarily shaped input waveforms, the main challenge of sequential cell timing analysis is to determine the conditions for the input signal to change the state of the output. For a typical latch design, this situation will take place when the capturing signal becomes inactive and its internal node has gone beyond the induced feedback loop's meta-stability point [11]. Translating this scenario to gate-level models, such as CSMs, is not trivial. Combinational cells can be easily modeled as their input fluctuations can be accurately translated to the output node, despite the fact that certain situations, such as multiple input simultaneous switching (MISS), can lead to significant errors [12]. On the contrary, feedback loops make the output node of sequential cells independent of the input nodes at the capturing signal inactive period, losing their characteristic input/output relationship, thus needing additional control mechanisms. Besides, unintended transitions in the stored value can be also considered as a MISS event, which CSMs are not able to model correctly. A CSM for sequential cells can be found in [8] where it is shown how a combinational CSM can be extended for sequential cells using a transmission gate based latch as an example. In particular, the cell is analyzed at its different modes of operation, extracting their respective CSMs and combining them into a quite complex CSM. Finally a D flip-flop CSM is also presented by connecting two complementary latches in series.

In this paper, we propose a general purpose statistical simulation engine for digital circuits which extends the previous work presented in [13,14] by including the analysis of sequential circuits. By using our statistical simplified transistor model (SSTM), a BSIM4-like transistor model [15], CSM's main limitations are avoided and higher accuracy is achieved at the expense of a slightly longer runtime. Additionally, since our simulation engine works at the transistor level, there is no difference in how sequential and combinational circuits are treated during the simulation. Sequential circuits are only treated in a special way when the initial guess of the DC solution is generated, due to their inherent combinational feedback loops. PV is captured by our SSTM by computing the different sensitivities to physical parameters for the selected process variables. A fast non-Monte Carlo statistical timing analysis method is used to find the statistical output arrival times by solving a system of random differential equations (RDE).

2 Simulation Engine

The typical work flow of any general purpose circuit simulator, like SPICE or SPECTRE, starts with the circuit description, as a text file listing all the discrete components within the circuit. This text file is then analyzed for correctness and translated into a mathematical representation. To perform transient analysis, the simulation engine must solve a system of linear ordinary differential equations (ODE) by successive discretization and linearization of the circuit's mathematical representation. DC analysis provides the circuit's initial operating point, thus ensuring a unique solution for the problem.

The proposed simulator, whose work flow is shown in Fig. 1, is composed of two main parts, a deterministic part and a statistical part. The deterministic part follows the simulation flow described before and, in this sense, is very similar to other SPICE-like circuit level simulators. On the other hand, the statistical part analyzes the circuit's response under PV, for which additional input data is required, similar to what a Monte Carlo loop would need. Our method calculates the sensitivities of the voltage waveforms, which are now characterized as stochastic processes, with respect to PV by solving a system of RDEs. Finally, the stochastic waveforms are processed to determine the statistical moments of the different timing parameters of interest like the crossing times at different voltage levels.

2.1 Statistical Simplified Transistor Model (SSTM)

The proposed circuit simulator uses a LUT-based statistical simplified transistor model (SSTM), which includes a voltage controlled current source (I_{ds}) and five non-linear parasitic capacitances (C_{gs}, C_{gd}, C_{gb}, C_{db} and C_{sb}). However, C_{db} and C_{sb} are further approximated as linear capacitors in view of their limited voltage dependence and relative small capacitance value. All these values are obtained running several DC SPECTRE simulations for both NMOS and PMOS transistors, using an accurate BSIM-4 transistor model [16].

The generated LUTs are accessed using the transistor voltages V_{gs} and V_{ds} (V_{sb} is also used for I_{ds} to take into account body biasing) as indices, in steps of 100 mV (50 mV for I_{ds} values). This fine grain voltage characterization allows us to use low order methods to compute off-grid values, such as bilinear interpolation (trilinear if body biasing) or 0-order extrapolation for out-of-bounds values.

Additional LUTs are constructed to capture the SSTM's parameters sensitivities to PV by first running the same characterization process for different values of the process parameters, like the transistor length, the oxide thickness or the threshold voltage. Finally, finite differences with respect to the each parameter nominal value are applied to obtain the sensitivities LUTs.

2.2 Circuit Description and MNA System Generation

A simplified SPICE-like netlist format has been defined for two different levels of abstraction (gate-level and transistor-level). The circuit is usually specified using the gate-level format, so the simulator's first step analyzes the circuit's topology and translates the gate-level format into the transistor-level using a simple parsing program.

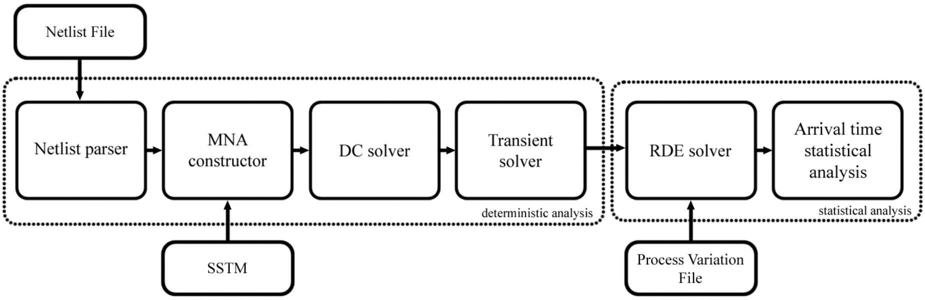


Fig. 1. Statistical simulator detailed overview

Being aware that non-linear devices are present within the circuit, an initial guess for the DC solution is computed. Since the circuit has been described using logic gates, this initial DC guess can be easily found using a breadth-first traversal algorithm along with basic boolean algebra, leaving only the gate's internal nodes as true unknowns. This algorithm works well if the circuit can be expressed as a directed acyclic graph (DAG) with logic gates and wires as vertices and edges respectively. Any combinational circuit is a clear example of these kinds of circuits. However, sequential circuits introduce combinational feedback loops, thus leading to cyclic graphs. To solve this problem, combinational feedback loops are identified and annotated during the circuit's topology analysis. This information is then used to resolve the loop's nodes initial values before the algorithm starts.

Modified nodal analysis (MNA) [17], the most common technique for systematic circuit analysis, is used by the simulator to set up the circuit's mathematical description. Here, the circuit's transistor-level description and the SSTM are combined to build the set of equations which define the circuit's behaviour, with output node voltages and device controlling branch currents as the system unknowns. Assuming inductor-less interconnection wires and output node voltages as the only system unknowns which need to be computed, the general MNA system is simplified, leading to the following system of ODEs:

$$\mathbf{C}(t)\dot{\mathbf{v}}(t) + \mathbf{G}\mathbf{v}(t) - \mathbf{i}(t) = \mathbf{0} \tag{1}$$

with $\mathbf{C}(t)$ the capacitance matrix, $\mathbf{G}$ the conductance matrix, $\mathbf{v}(t)$ and $\dot{\mathbf{v}}(t)$ the voltages and their time derivatives, and $\mathbf{i}(t)$ the input current sources.

In practice, only systems of linear ODEs can be efficiently solved by a computer program. However, the system in (1) contains non-linear elements introduced by the SSTM in $\mathbf{C}(t)$ and $\mathbf{i}(t)$. The general approach to solve such systems is to approximate every non-linear component by a linear equivalent and iterate, using a Newton-like algorithm, until the approximation error is small enough. To handle this problem, our simulator separates each of these matrices into a constant value part and a time dependent part. While the former is kept in a matrix format, the latter is stored in a transistor indexed data structure. During the DC and transient analysis the linearized system is constructed for each time instant and Newton-like iteration by extracting, from each transistor data structure, the required values of their non-linear parameters. In this sense, Jacobian matrices, needed for Taylor expansion of non-linear elements, are also stored in the same way.

2.3 DC Solver and Transient Analysis

The MNA system in (1) is further simplified by removing the time dependent terms and, along with the initial guess found during the gate-level translation, is now used to find the circuit's exact DC solution, yielding the following system of equations in $\mathbf{V}_0 = \mathbf{v}(t_0)$:

$$\mathbf{G}\mathbf{V}_0 - \mathbf{I}_0 = \mathbf{0} \tag{2}$$

The system in (2) is still non-linear due to the transistors' current source contribution to the output nodes. Therefore, a Newton-like iterative method can be used to solve this system, provided that the initial guess is close enough to the final solution. Although this is usually true for our computed initial guess, additional convergence strategies are used to ensure the algorithm is able to find a solution. In particular, our simulator adds large resistors to ground at every output node to deal with potentially isolated nodes due to non-linear devices and limits node voltage variations from two consecutive iterations thus solving the problem of non-convergent oscillating solutions (known as *Gmin* and *voltage damping* respectively).

The circuit's time response under nominal process conditions, $\mathbf{p_0}$, can be found by solving the MNA system described in (1) with the initial conditions obtained in (2). The resulting initial value problem can be rewritten as:

$$F(\dot{\mathbf{v}}, \mathbf{v}, t, \mathbf{p_0}) = \mathbf{0}, \text{ with } \mathbf{v}(t_o) = \mathbf{V_0} \tag{3}$$

To find the solution to this system of non-linear ODEs, an implicit linear multistep (LMS) method with variable time step (t_{step}) is used, based on a simple *predictor-corrector* method. Polynomial extrapolation is first used to calculate an initial guess for the solution at the new time instant $t_{k+1} = t_k + t_{step}$ (prediction) while a Newton-like iterative algorithm along with a the desired LMS method (*backward Euler, BE* or *trapezoidal rule, TR*), which can be chosen at the beginning of the simulation, is then used as the correction method. If convergence is achieved in a limited number of iterations, Milne's principal local truncation error (PLTE) estimate is computed for accuracy check and a new time instant and t_{step} are decided upon this. Otherwise the predicted solution is rejected and t_{step} is reduced for a new loop iteration [18].

2.4 Statistical Solver and Crossing Time Statistical Analysis

As a consequence of PV affecting the different circuit's elements, the resulting voltage waveforms become stochastic processes. Under these new conditions, Equation (3) becomes a system of non-linear RDEs:

$$F(\dot{\mathbf{v}}, \mathbf{v}, t, \mathbf{p}) = \mathbf{0}, \text{ with } \mathbf{v}(t_o) = \mathbf{V_0} + \boldsymbol{\delta}_{V_0} \tag{4}$$

In (4), $\mathbf{p}$ represents the vector of PV, expressed as $\mathbf{p} = \mathbf{p_0} + \boldsymbol{\xi}$, with $\boldsymbol{\xi}$ the vector of random deviations of the process parameters from the nominal conditions, a vector of random variables with zero mean and σ standard deviation. Finally $\boldsymbol{\delta}_{V_0}$ represents the deviation of the initial conditions due to PV.

Intuitively, the solution of (4) will be close to the deterministic solution found for (3), $\mathbf{v}_n(t)$ [13]. Therefore, if small deviations are assumed, a first order Taylor expansion around $\mathbf{v}_n(t)$ is a valid approximation of (4), resulting in a system of linear RDEs in the new variable $\mathbf{y}(t) = \mathbf{v}(t) - \mathbf{v}_n(t)$, the voltage deviation from the nominal solution:

$$\mathbf{M}(t)\dot{\mathbf{y}}(t) + \mathbf{R}(t)\mathbf{y}(t) + \mathbf{Q}(t)\boldsymbol{\xi} = \mathbf{0} \tag{5}$$

with $\mathbf{M}(t)$, $\mathbf{R}(t)$ and $\mathbf{Q}(t)$ the partial derivatives of F with respect to $\dot{\mathbf{v}}$, $\mathbf{v}$ and $\mathbf{p}$ respectively. A system like this has a unique solution in the mean square [19]:

$$\mathbf{y}(t) = \boldsymbol{\Phi}(t, t_0)\mathbf{y_0} - \int_t^{t_0} \boldsymbol{\Phi}(t, u)\mathbf{Q}(u)\boldsymbol{\xi}\, du = \boldsymbol{\alpha}(t)\boldsymbol{\xi} \tag{6}$$

with $\boldsymbol{\Phi}(t, t_0)$ the solution of the equivalent homogeneous system. As can be seen in (6) the output voltage deviations $\mathbf{y}(t)$ are proportional to $\boldsymbol{\xi}$, with $\boldsymbol{\alpha}(t)$ the sensitivities of the voltage deviation waveforms with respect to $\boldsymbol{\xi}$. Using this relationship, the equivalent system of linear ODEs in $\boldsymbol{\alpha}(t)$ is constructed from (5). The simulator can solve this system by using simplified TR integration method, and the stochastic voltage waveforms can finally be expressed as:

$$\mathbf{v}(t) = \mathbf{v}_n(t) + \mathbf{y}(t) = \mathbf{v}_n(t) + \boldsymbol{\alpha}(t)\boldsymbol{\xi} \tag{7}$$

Finally, to perform timing analysis, the circuit delay is computed as the difference between the voltage crossing times, t_η. These crossing times are defined as the time instant each signal reaches a target voltage value, usually expressed as a percentage of the supply voltage. In the presence of PV, the voltage crossing time of a signal $v(t)$ is also a random variable which can be expressed in a similar way as (7):

$$t_\eta = t_{\eta_{NOM}} + \boldsymbol{\beta}_{t_\eta}\boldsymbol{\xi}, \text{ with } \boldsymbol{\beta}_{t_\eta} = \frac{\partial t_\eta}{\partial \boldsymbol{\xi}} \tag{8}$$

For small deviations, the sensitivity vector $\boldsymbol{\beta}_{t_\eta}$ can be approximated with its value at the nominal crossing time $t_{\eta_{NOM}}$ of the signal $v(t)$ as follows [20]:

$$\left.\frac{dv(t)}{d\boldsymbol{\xi}}\right|_{t=t_{\eta_{NOM}}} = \left.\left(\frac{\partial v_n(t)}{\partial t} \times \frac{\partial t}{\partial \boldsymbol{\xi}}\right)\right|_{t=t_{\eta_{NOM}}} + \boldsymbol{\alpha}(t_{\eta_{NOM}}) = \mathbf{0} \tag{9a}$$

$$\boldsymbol{\beta}_{t_\eta} \approx \left.\frac{\partial t}{\partial \boldsymbol{\xi}}\right|_{t=t_{\eta_{NOM}}} = -\frac{\boldsymbol{\alpha}(t_{\eta_{NOM}})}{\left.\frac{\partial v_n(t)}{\partial t}\right|_{t=t_{\eta_{NOM}}}} \tag{9b}$$

3 Experimental Results

As we pointed at the beginning of this paper, our main concern is the accurate timing analysis of sequential circuits under PV using our non-Monte Carlo statistical simulator. In order to test our simulator accuracy, three different sequential circuits, with an increasing level of complexity, were selected: *i*) a high level-active transparent latch (DLH_X1, 16 transistors); *ii*) a positive-edge master-slave D flip-flop (DFF_X1, 28 transistors); and *iii*) a custom sequential circuit (SEQ_X1, 90 transistors). The first two circuits are an obvious choice since they are the most common sequential elements used in synchronous designs. The last one, shown in Figure 2, tries to recreate a more realistic scenario with launching and catching flip-flops, combinational logic and a more elaborated wire model rather than a simple capacitor to ground, which introduces

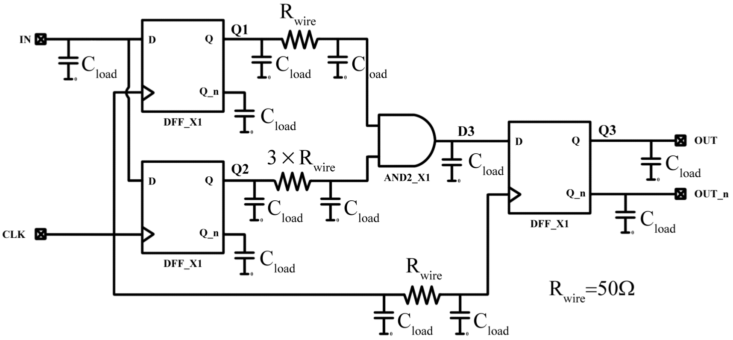


Fig. 2. Sequential test circuit (SEQ_X1)

non-zero skew in the clock network. In addition to this, MISS is also present at the circuit's combinational gate since its input signals have similar delays and are captured at the same clock edge. All the circuits have been built using the Nangate 45 nm Open Cell Library as reference [16]. Figure 3 shows the internal structure of the tested D-flip-flop. Details about transistor sizing can be found at the library documentation.

We analyzed the proposed test circuits using our statistical simulator, implemented in MATLAB, and comparing the results against BSIM4-based SPECTRE 10 K Monte Carlo simulations. This experiment was repeated for different values of load capacitance, ranging from 5 to 25 fF, and 100 ps transition time piecewise linear functions as input signals. Transistor's length (L) and threshold voltage (V_{th}) were modeled as global non-correlated normal distributions to simulate PV with 0.5 nm and 0.04 V standard deviations from their nominal values respectively. Trapezoidal rule (TR) integration method was used to ensure the best possible accuracy.

Table 1 shows the 50% delay mean and standard deviation relative errors at the most important nodes for each test circuit nodes. From these results, we can see that the mean error is, in most of the cases, below 1% and it gets closer to zero as the load capacitor grows. The worst mean error values are found for the transparent latch DLH_X1, being a consequence of the charge non-conserving

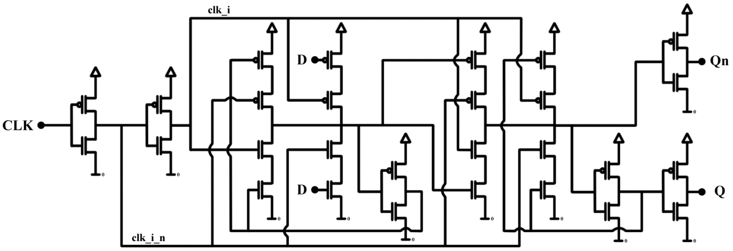


Fig. 3. D flip-flop schematic (DFF_X1)

Table 1. 50% delay statistical analysis under PV (rising input data signal)

PV	Load capacitance (fF)					Load capacitance (fF)				
	5	10	15	20	25	5	10	15	20	25
	Mean (μ) rel. error (%)					Std. deviation (σ) rel. error (%)				
DLH_X1's OUTPUT NODE (Q) $T_{CLK\text{-}O}$										
L	-3.37	-2.29	-1.66	-1.34	-1.10	-9.27	-6.46	-6.31	-4.48	-4.59
V_{th}	-3.86	-2.87	-2.26	-1.97	-1.75	-13.78	-3.16	-1.88	-1.84	-0.83
L & V_{th}	-3.85	-2.86	-2.27	-1.95	-1.75	-12.74	-5.28	-2.28	-2.76	-1.57
DFF_X1's OUTPUT NODE (Q) $T_{CLK\text{-}O}$										
L	-0.54	-0.37	-0.37	-0.25	-0.16	-7.85	-6.99	-6.68	-3.46	-4.37
V_{th}	-1.05	-0.96	-1.00	-0.89	-0.81	5.07	1.88	1.83	2.05	0.07
L & V_{th}	-1.04	-0.95	-0.49	-0.88	-0.79	-1.04	-0.49	-0.53	1.72	-0.28
SEQ_X1's 1st LAUNCHING DFF OUTPUT NODE (Q1) $T_{CLK\text{-}O}$										
L	0.08	0.04	0.03	0.02	0.02	-7.15	-5.34	-4.42	-4.10	-4.24
V_{th}	-0.61	-0.68	-0.71	-0.72	-0.72	1.35	-0.08	-0.42	-0.26	-0.36
L & V_{th}	-0.59	-0.67	-0.69	-0.70	-0.71	-1.04	-0.88	-0.81	-0.49	-0.83
SEQ_X1's CATCHING DFF INPUT NODE (D3) $T_{50\%}$										
L	0.13	0.18	0.20	0.20	0.20	-6.18	-5.86	-6.27	-6.71	-6.31
V_{th}	-0.19	-0.15	-0.13	-0.13	-0.12	-6.80	-8.13	-8.86	-9.11	-10.14
L & V_{th}	-0.18	-0.15	-0.13	-0.12	-0.12	-7.42	-8.08	-8.43	-9.03	-9.45
SEQ_X1's CATCHING DFF OUTPUT NODE (Q3) $T_{CLK\text{-}O}$										
L	-0.51	-0.48	-0.48	-0.48	-0.54	-7.45	-6.33	-5.53	-4.68	-3.40
V_{th}	-1.06	-1.14	-1.18	-1.20	-1.23	5.13	2.68	2.79	1.34	2.41
L & V_{th}	-1.05	-1.12	-1.16	-1.18	-1.20	-0.15	1.07	0.79	0.75	0.62

transistor model for the non-linear parasitic capacitors and it gets reduced as the linear load capacitance grows. Regarding the standard deviation, the results also show a decreasing trend with higher load values for most of the analyzed circuits although the relative error here is significantly larger. Again charge non-conserving capacitive models are the main source of error. Finally, the runtime improvement achieved with our method compared with Monte Carlo simulation, is quite significant, as can be seen in Table 2. However, the current simulation implementation has still some complexity problems dealing with large circuits mainly due to the fact that sparse matrix techniques have not been applied.

Table 2. Runtime comparison of the different simulation methods

Simulation method	DLH_X1	DFF_X1	SEQ_X1
MATLAB	25 secs.	45 secs.	12 mins.
SPECTRE 10 K MC	1800 secs.	2400 secs.	120 mins.

4 Conclusion

In this paper we have presented a general purpose statistical circuit simulator for accurate timing analysis, which is mandatory for state of the art integrated circuits verification where random deviations of physical parameters play a relevant role in the circuit's behaviour. A statistical simplified transistor model, instead of gate-level models, along with a fast non-Monte Carlo statistical method allow us to accurately simulate any input circuit, thus overcoming the limitations of gate-level models regarding sequential cells, as can be seen in the conducted experiments.

References

1. Hitchcock, R.B., Smith, G.L., Cheng, D.D.: Timing Analysis of Computer Hardware. IBM Journal of Research and Development, 100–105 (1982)
2. Hitchcock, R.B.: Timing Verification and the Timing Analysis program. In: Proc. of the DAC, pp. 594–604 (1982)
3. Blaauw, D., Chopra, K., Srivastava, A., Scheffer, L.: Statistical Timing Analysis: From Basic Principles to State the Art. IEEE Trans. on CAD of Integrated Circuits and Systems, 589–607 (2008)
4. Berkelaar, M.: Statistical Delay Calculation, a Linear Time Method. In: Proc. of TAU, pp. 15–24 (1997)
5. Visweswariah, C., Ravindran, K., Kalafala, K., Walker, S.G., Narayan, S.: First-Order Incremental Block-Based Statistical Timing Analysis. In: Proc. of the DAC, pp. 331–336 (2004)
6. Agarwal, A., Blaauw, D., Zolotov, V.: Statistical Timing Analysis for Intra-Die Process Variations with Spatial Correlations. In: Proc. of the IEEE/ACM ICCAD, pp. 900–907 (2003)
7. Croix, J.F., Wong, D.F.: Blade and Razor: Cell and Interconnect Delay Analysis Using Current-Based Models. In: Proc. of the DAC, pp. 386–389 (2003)
8. Nazarian, S., Fatemi, H., Pedram, M.: Accurate Timing and Noise Analysis of Combinational and Sequential Logic Cells Using Current Source Modeling. IEEE Trans. Very Large Scale Integrated Systems, 92–103 (2011)
9. Fatemi, H., Nazarian, S., Pedram, M.: Statistical Logic Cell Delay Analysis Using a Current-based Model. In: Proc. of the DAC, pp. 253–256 (2006)
10. Salman, E., Dasdan, A., Taraporevala, F., Kucukcakar, K., Friedman, E.G.: Exploiting Setup–Hold-Time Interdependence in Static Timing Analysis. IEEE Trans. on CAD of Integrated Circuits and Systems, 1114–1125 (2007)
11. Oh, N., Ding, L., Kasnavi, A.: Fast Sequential Cell Noise Immunity Characterization Using Meta-stable Point of Feedback Loop. In: Proc. of the ISQED, pp. 153–159 (2006)
12. Amin, C., Kashyap, C., Menezes, N., Killpack, K., Chiprout, E.: A Multi-port Current Source Model for Multiple-Input Switching Effects in CMOS Library Cells. In: Proc. of the DAC, pp. 247–252 (2006)
13. Tang, Q., Zjajo, A., Berkelaar, M., van der Meijs, N.: RDE-Based Transistor-Level Gate Simulation for Statistical Static Timing Analysis. In: Proc. of the DAC, pp. 787–792 (2010)

14. Tang, Q., Zjajo, A., Berkelaar, M., van der Meijs, N.: Transistor-level gate model based statistical timing analysis considering correlations. In: DATE, pp. 917–922 (2012)
15. UC Berkeley. BSIM4 MOSFET Model (2003), http://www-device.eecs.berkeley.edu/bsim/?page=BSIM4
16. Nangate 45nm Open Cell Library (2009), http://www.nangate.com/?page_id=22
17. Ho, C.W., Ruehli, A., Brennan, P.: The Modified Nodal Approach to Network Analysis. IEEE Transactions on Circuits and Systems, 504–509 (1975)
18. Najm, F.N.: Circuit Simulation. Wiley-IEEE Press (2010)
19. Soong, T.T.: Random Differential Equations in Science and Engineering. Academic Press, New York (1973)
20. Tang, Q.: Personal communication

PVTA Tolerant Self-adaptive Clock Generation Architecture

Jordi Pérez-Puigdemont[1], Antonio Calomarde[2], and Francesc Moll[1]

[1] Dept. of Electronic Engineering, Universitat Politècnica de Catalunya,
Barcelona, Spain
jordi.perez-puigdemont@upc.edu

[2] Dept. of Electronic Engineering, Universitat Politècnica de Catalunya,
Terrassa, Spain

Abstract. In this work we propose a self-adaptive clock based on a ring oscillator as the solution for the increasing uncertainty in the critical path delay. This increase in uncertainty forces to add more safety margins to the clock period which produces a circuit performance downgrade. We evaluate three self-adaptive clock systems: free running ring oscillator, infinite impulse response filter controlled RO and TEAtime controlled ring oscillator. The safety margin reduction of the three alternatives is investigated under different clock distribution delay conditions, dynamic variation frequencies and the presence of mismatch between the ring oscillator and the critical paths and the delay sensors.

1 Introduction

Modern digital systems rely on synchronous circuit architectures. On any synchronous circuit the clock is the most critical signal and its period is a critical parameter that has to be carefully selected. The clock period has to be long enough to accommodate the critical path (CP) delay plus the set-up time and the clock-to-output delay of the registers. Since there is an uncertainty component in the delay of every logic gate due to the process, voltage, temperature and aging (PVTA) variations a safety margin has to be added to the clock period. This safety margin ensures a correct operation of the synchronous system. The more margin added, the more unlikely to fail the chip is. However, the introduction of safety margins (SM) represents a loss in performance. Alternatively, if it is possible, SM can be added to the supply voltage instead of to the clock period. In this case the yield is increased but at the price of more power consumption.

PVTA variations can be classified as static or dynamic and spatially homogeneous or heterogeneous. Table 1 classifies the most common variations following this taxonomy.

The margin added to the clock period or supply voltage has to be carefully determined. PVTA variations produce a delay uncertainty which is hard or, in some cases, impossible to predict. Different techniques like corner analysis, SSTA, etc; are used to estimate the safety margin that, once added to the clock period or the supply voltage, produces a desired yield.

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 142–154, 2013.

Table 1. Common sources of variability classified by its temporal, static or dynamic, and spatial, homogeneous or heterogeneous, behaviour

	Static	Dynamic
Homogeneous	– Die to die (D2D) process variations.	– Voltage regulation module (VRM) ripple. – Room temperature variations. – Off chip voltage drops.
Heterogeneous	– Within die (WID) process variations. – Device to device random (RND) process variations.	– Simultaneous switching noise (SSN). – IR drop. – Temperature hotspots. – Ageing.

As the transistors minimum size shrinks, the uncertainty due to process variations increases as well as the aging effects become more important [1, 2]. In addition, the transistor size reduction makes possible more complex circuits. This complexity increase leads to an escalation in the number of possible CPs. More CPs impose a larger SM to satisfy a given yield [3].

Smaller transistors facilitate to integrate more functionalities in the same die, increasing the power demand variability. This uncertainty in the consumed power by the circuit blocks makes more difficult to estimate the supply voltage variation such as IR drop or simultaneous switching noise. Also, transistor miniaturization may permit the integration of voltage regulator modules on the die. This integration step is expected to induce more supply voltage ripple than from off-die regulators [4].

Also, the temperature of the circuit can vary depending on the computation carried out since the amount of demanded current by its different blocks depends on the executed instructions. On top of this the temperature also depends on the temperature of the environment where the chip operates.

As the amount of uncertainty reaches its highest value and its estimation during the design stage consumes more and more resources, a new paradigm in the synchronous circuit design is needed. Some authors had proposed the adaptation of the clock period to PVTA variations [5, 6]. We propose in this article a new approach in this field: the self-adaptation of the clock period to ensure the correct operation of any synchronous circuit under PVTA variations.

In section 2 we show the natural capability of ring oscillators (RO) to act as self-adaptive clock sources that can cope with PVTA variations. Also, in this section, its weaknesses are revised. In section 3 a closed loop control architecture for the ring oscillator is proposed in order to cope with the RO weaknesses. In section 4 the simulation results are presented and the advantages and disadvantages of the closed loop controlled RO in front of a free running ring oscillator and an increment controlled RO discussed. And finally, in section 5 the conclusions are exposed.

2 RO Adaptation to PVTA Variations

A ring oscillator (RO) is an oscillating circuit: a chain of inverting and non inverting stages, where the number of inverting stages is odd and the chain output is connected to the chain input.

ROs, due to its oscillating nature, can be used to generate clock signals but, in digital systems, they are not used to carry out this duty because its high sensitivity to PVTA variations. This high sensitivity is normally accounted as a source of clock period indetermination.

Contrary to be an undesired effect, the RO sensitivity to PVTA variations can be the key point to build a clock signal source where its period is adapted to the circuit environment conditions. This change of perspective will lead us to stop relying on fixed clock signals like PLLs and reduce the safety margins added, during the design stages, to the clock period and/or the supply voltage. As a side effect, the resources and time spent estimating PVTA effects on the CP delay during the design stages will be also reduced.

Let us assume an ideal case in which the RO suffers the same PVTA variations as all the candidates to operate as a CP and the clock distribution is instantaneous (Fig. 1). Under these naive operating condition assumptions it is obvious that the RO generated clock will adapt its period to the instantaneous delay suffered by the gates in the die. In this way the SM can be reduced with respect to a fixed clock, which period is independent of process and operation conditions. Unfortunately these conditions do not take place in reality.

2.1 RO Limitations

The limitations of the RO clock generation are caused by the mismatch between the PVTA variations that take place in the RO gates and all the other gates circuit, specially the CP.

This mismatch can be caused by the heterogeneity of the variations along the die such as within die variations (WID) due to process, different IR drops on V_{dd}, temperature differences in the chip, etc. Also the mismatch can arise if the variations are homogeneous but have a dynamic component. The clock generated by the RO need to be distributed all along the die through a clock distribution network (CDN). The CDN imposes a delay, t_{clk}, between the generated clock and the delivered clock signals. The period of the delivered clock, at the end of

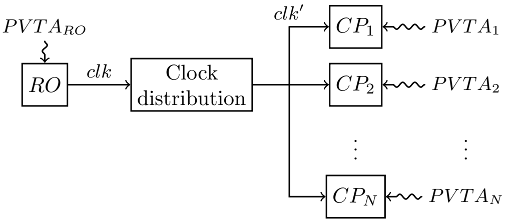


Fig. 1. Free running ring oscillator (RO) used as a self-adaptive clock generator. The main elements that undermine the performance of RO as a self-adaptive clock generator are depicted: the mismatch between the variations suffered by the RO and the circuit critical paths (CP); and the clock distribution network which introduces a delay between the generated and the delivered clock signal to the CPs.

the CDN, will be adapted to the variations that occur t_{clk} before, not at that instant.

Against heterogeneous variations, static or dynamic, the RO can fail reducing the safety margin. The RO circuit adapts its period to the environment conditions around it. In fact, the RO acts like a point sensor.

3 Closed Loop Controled RO

To cope with the spatial heterogeneity of the PVTA variations we propose to disseminate sensors all over the clock domain. As sensors we propose the time digital converter (TDC) [7]. TDC outputs, every clock cycle, the number of crossed gates, or stages, by an alternating signal during the last period. This integer number give us a sense of the delay suffered by the gates near each TDC. If the output of the TDC is low means that the logic gates are experimenting an extra delay due to the variations; or vice-versa when the output is high. The stages of the TDCs and the RO are supposed to be equal. This equivalence will not take place in reality, for this reason we will have to take into account heterogeneous static and dynamic variations.

Once we have some sensors on the core we can compare, at each period, the worst sensor output τ, this is the lowest output among all the TDCs, with a given set-point c and then take some actions over the RO, *i.e.* changing its length l_{RO}, in order to adapt the clock period to the variations that the RO can not sense. This closed loop controlled RO architecture is depicted in Fig. 2. τ is related to the logic depth a signal can traverse in one period given PVTA conditions in the area around the TDC. When $\tau < c$ means that the period is too short and a logic error may occur. When $\tau > c$ no error occurs, but there is a potential loss in performance since the clock period is too large for the given PVTA condition.

By having a clock managed by a closed control loop with a set-point input, the clock period needs not to be set during the design stage, just the minimum and maximum number of RO stages. This leads to a more relaxed CPs delays estimation. Once the chip is produced and it is running, we only need to choose

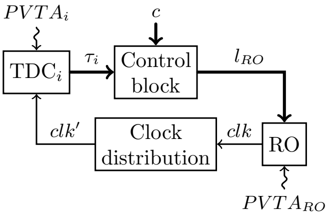


Fig. 2. Closed loop controlled ring oscillator architecture. The main signals are labelled: the set-point c which is the only input of the clock generation system, the ring oscillator length l_{RO}, the generated clock clk, the distributed clock clk' and the i-th TDC lecture τ_i. Also we point out that the clock generator, the RO, and the sensors, the TDCs, could suffer different variations.

the correct set-point c that allows the system to run without any error and/or maximizes the computation throughput. Therefore the pipeline needs, at least, error detection capacities.

In this article, the delay and the period are expressed in number of stages. In fact the units of c, l_{RO} and τ_i are number of stages. Set-point c is the desired output from the TDCs τ_i and l_{RO} is the length of the RO.

Since every event is triggered by the clock edges the architectural view in Fig. 2 can be translated into a discrete control system view as shown in Fig. 3. As a first approximation to the problem we modelled the action of the RO, CDN and TDC as a simple delay chain with the addition of perturbations, that account for the heterogeneous and homogeneous variations.

When the RO length l_{RO} is changed the clock period changes the value of its period in the next clock period. Then this clock has to be distributed through the CDN and will take M periods to arrive to the registers. The value of M will depend on the period of the clock signal $T_{clk}[n]$ at each step and the delay of the CDN t_{clk}: $M[n] = \lceil t_{clk}/T_{clk}[n] \rceil$. Once the clock arrives to the registers the TDCs outputs the number of crossed stages τ during the last period. τ is compared with the set-point c in order to generate a error value $\delta = c - \tau$. δ is injected into the control filter $H(z)$ that will, after one period, give the new l_{RO}.

The period of the clock generated by the RO can be influenced by an homogeneous variation e, which affects equally the TDCs, but also by an heterogeneous variation μ which will be different to the variations that takes place at the different TDCs. When the same variation affects the RO and the TDC the output of the TDC does not vary. Therefore, in the discrete control system schema (Fig. 3), the perturbations in the TDC and in the RO present opposite sign.

3.1 Control Block Constraints

Once the control loop is defined it is possible to find out how the two most important magnitudes, δ and l_{RO}, behave when some change occur in the perturbations, *i.e.* e and/or μ, or set-point, c.

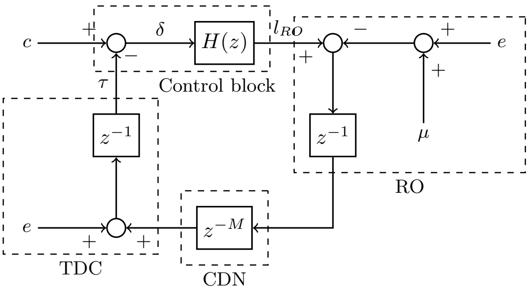


Fig. 3. Discrete system view of the closed loop controlled ring oscillator. The main blocks of the architecture are labelled. The system inputs are labelled: the set-point c, homogeneous variation e and heterogeneous variation μ. As well as important signals: RO length l_{RO} and adaptation error δ. The CDN number of samples M delay depends on the input value to the CDN block and the CDN delay in number of stages: $M = \lceil t_{clk}/T_{clk} \rceil$.

As depicted in Fig. 3 we can derive, in the z-domain, the l_{RO} and δ expressions as function of the combined inputs $p(z)$, assuming that the control block transfer function is $H(z) = N(z)/D(z)$:

$$H_{l_{RO}}(z) = \frac{l_{RO}(z)}{p(z)} = \frac{N(z)}{D(z) + N(z)z^{-M-2}} \tag{1}$$

$$H_{\delta}(z) = \frac{\delta(z)}{p(z)} = \frac{D(z)}{D(z) + N(z)z^{-M-2}} \tag{2}$$

where $p(z) = c(z) + e(z)\left(1 - z^{-M-1}\right)z^{-1} - \mu(z)z^{-M-2}$.

If we assume that $p(z)$ is a Heaviside step, when $t \to \infty$, the desired value for δ and l_{RO} respectively are:

$$\lim_{t\to\infty} h_{l_{RO}}(t) * u(t) \neq 0 \tag{3}$$

$$\lim_{t\to\infty} h_{\delta}(t) * u(t) \quad = 0 \tag{4}$$

this is that under a minimum perturbation the value of l_{RO} changes to counteract it (3) and, consequently, the error value δ tends to zero (4). Using the final value theorem we can re write (3) and (4) as (5) and (6) respectively:

$$\lim_{z\to 1} (z-1)H_{l_{RO}}(z)U(z) \neq 0 \tag{5}$$

$$\lim_{z\to 1} (z-1)H_{\delta}(z)U(z) \quad = 0 \tag{6}$$

which lead us to a a set of constraints of $N(z)$ and $D(z)$:

$$N(z)|_{z=1} \neq 0 \tag{7}$$

$$D(z)|_{z=1} = 0 \tag{8}$$

3.2 Control Block Implementations

In this section we propose two different implementations of the control filter $H(z)$. The first is an infinite impulse response (IIR) filter and, the second, a TEAtime implementation similar to the proposed in [8, 9].

The IIR control block architecture proposed is depicted in Fig. 4. It is slightly different to a standard IIR filter due to the constraints found in sec. 3.1 and some implementation constraints, like the aim of reducing the clock generation circuit area overhead. Due to this we choose to operate over the integers avoiding the use of floating point operations. Secondly the gain values all along the IIR control block are constrained to powers of two in order to simplify multiplication operations. Since we choose to operate over the integers we need to minimize the rounding error inside the filter. To do so, we scale the signal (k_{exp} and ${k_{exp}}^{-1}$). Another difference with common IIR filters is the k^* gain (Fig. 4) added to ensure the fulfilment of constraints (7) and (8) when the IIR has more than one coefficient. Also we added an extra delay after k^* gain in order to take into account the possible need of a large adder to implement the control block that could be necessary to pipeline it.

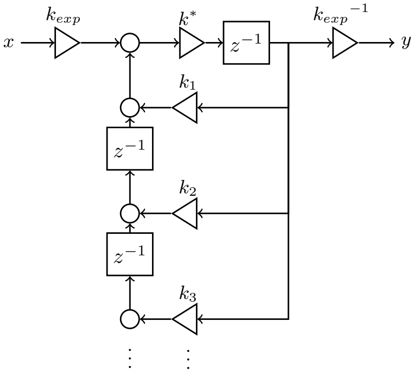


Fig. 4. IIR control block implementation proposal

The transfer function of the proposed IIR filter (Fig. 4) is the following:

$$H_{IIR}(z) = \frac{z^{-1}}{\dfrac{1}{k^*} - \displaystyle\sum_{i=1}^{N} k_i z^{-i}} \tag{9}$$

To fulfil the constraints 7 and 8 the filter coefficients have to follow the next relation:

$$k^* = \left(\sum_{i=1}^{N} k_i \right)^{-1} \tag{10}$$

For the TEAtime implementation, the control block $H_{TEA}(z)$, is depicted in Fig. 5. In this case there are no parameters to set and therefore the constraints do no apply in this case.

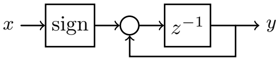


Fig. 5. TEAtime control block implementation inspired from [8, 9]. Where sign block performs the signum function.

4 Architecture Simulation

To perform the simulations we used the Simulink® software from Mathworks®. We simulated the adaptive response of three different systems: the proposed IIR controlled RO, a TEAtime controlled RO and a free running RO.

The chosen gain parameters for the IIR controlled RO are: $k_{exp} = 8$, $k^* = 1/2$, $k_1 = 1$, $k_2 = 1/2$, $k_3 = 1/4$, and $k_4 = k_5 = 1/8$. With these values we achieve a balance between filter adaptation velocity and low output ripple. k_{exp} value is chosen to ensure that the minimum perturbation propagates through all the branches of the filter. The set-point value for all the simulations is $c = 64$, this is the desired TDCs reading. In this simulations we have taken into account the quantization effects of the filter since, inside the it, we had only allowed integer number operations. We have used a high enough number of bits to represent the signals in order to avoid saturation, therefore we do not investigate the effects of signal saturation. Signal saturation influence on the adaptive clock will be studied in further work. The amplitude of the periodic perturbation e is set equal to $0.2c$, this is a 20% homogeneous variation.

4.1 Homogeneous Dynamic Variation (HoDV)

In Fig. 6 the timing error $\tau - c$ due to a HoDV of different frequencies is shown for a CDN delay equal to c stages. In the top plot the period of the perturbation is equal to $25c$ stages, that is 25 times the nominal period value. In this plot is possible to see that the negative timing error which is equal, in absolute value, to the needed safety margin, is quite close to the margin that would need a fixed clock to assure a error free operation, nevertheless the amplitude of the timing error is reduced.

In the middle plot of Fig. 6 the period of the perturbation is augmented to $37.5c$ stages. As the period of the perturbation is augmented the system can adapt the clock frequency better and, consequently, reduce the impact of the HoDV.

And finally, in the Fig. 6 lower plot, the period of the perturbation is increased to $50c$ stages. Once the perturbation is low enough its impact on the timing error $\tau - c$ gets even more reduced for the three adaptive clock systems here considered.

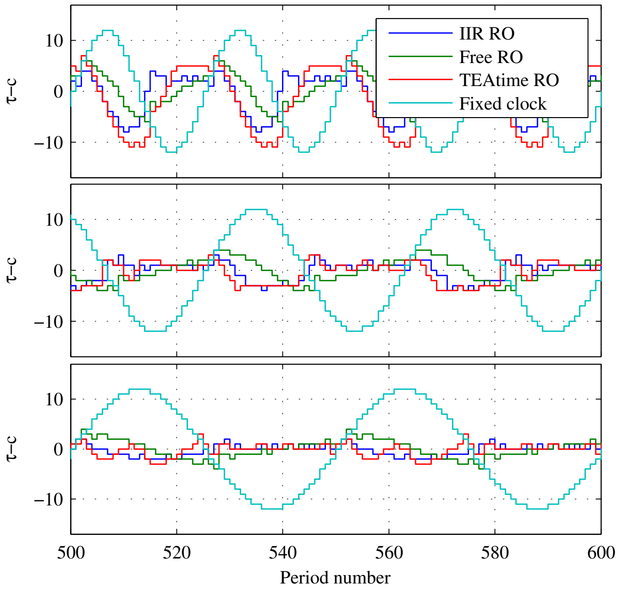


Fig. 6. Timing error $\tau - c$ for different clock generation systems. For the three plots the parameter are: set-point $c = 64$, HoDV amplitude equal to 20% of c, and the clock distribution delay equal to one clock period, this is equal to c stages. No mismatch between RO and TDC has been introduced. Upper plot: the period of the perturbation is set to $25c$. In this case the safety margin is slightly reduced with respect to a fixed clock. Middle plot: the perturbation period is set to $37.5c$. An appreciable adaptation error reduction takes place once the perturbation frequency is decreased. Lower plot: the perturbation period is set to $50c$. The Adaptation error is reduced to a minimum value.

In Fig. 6 is possible to see that the different adaptive clock generation systems evaluated can reduce the timing error due to a HoDV but it is hard to say which one achieves the greatest reduction, that is the best adaptation. In order to evaluate adaptive clock systems in which the period changes with the PVTA, we need to compare the mean clock period when no errors are detected. For this reason, we use as figure of merit the relation between the mean clock period of the adaptive clock to the fixed clock period, this is the relative adaptive period $\langle T_{clk} \rangle / T_{clk\,fixed}$.

In Fig. 7 $\langle T_{clk} \rangle / T_{clk\,fixed}$ is shown for different scenarios under a HoDV. In Fig. 7 upper plot the period of the perturbation is kept fixed, $T_e = 100c$, and the CDN delay is changed. Until $t_{clk}/c = 5$ IIR controlled RO is slightly the best option against the free running RO. In Fig. 7 lower plot the CDN delay is kept constant and the period of the perturbation is varied. Here the free running RO seems to be the best option under almost any situation. Also the free RO is the first adaptive system to cross the $\langle T_{clk} \rangle / T_{clk\,fixed} = 1$ boundary. Below which the use of adaptive systems makes sense since the safety margin is reduced.

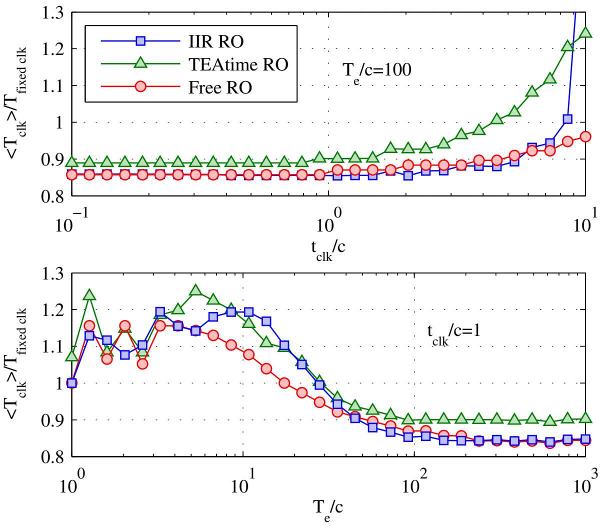


Fig. 7. Relative adaptive period for three different adaptive control systems under HoDV. Upper plot: The variation period is kept constant, $T_e = 100c$, and the CDN delay is varied: until $t_{clk}/c = 5$ the IIR controlled RO show the best performance, just slightly better than the free running RO, which turns to be the best option for $t_{clk}/c > 5$. Lower plot: The CDN delay is kept constant ($t_{clk} = c$) and the period of the perturbation is varied: the free running RO shows the best adaptation capacities on almost the whole T_e/c range. Free running RO is the first adaptive system to obtain a reduction of the safety margin ($\langle T_{clk} \rangle / T_{clk\,fixed} \leq 1$).

To conclude with an example, the HoDV adaptation results can be translated in terms of period measured in seconds. Let us assume that the set-point $c = 64$ generates, in ideal conditions, a clock period $T_{clk} = 1$ns. Under a CP delay variation up to 20% the clock period has to be set to $T_{clk} = 1.2$ns, or in the number of stages nomenclature the set-point should be changed to $c = 77$. Also assume that the adaptive clock allows to reduce the needed c, which assures an error free operation, up to 10%. This can be translated as a reduction of 0.12ns in the clock period, which is a 60% reduction of the added SM.

4.2 Heterogeneous Dynamic Variation (HeDV)

In Fig. 7 it was shown that under different conditions the free running RO is the best option to cope with HoDV. But as we consider the HeDV, introduced through a mismatch offset μ as indicated in Fig. 3, the best adaptive clock generation system option will not be the free RO any more. In Fig. 8 the relative adaptive period, for different period of the perturbation and CDN delay scenarios, is shown when there is a mismatch, up to 20%, μ between the RO and the TDC which are also under a HoDV. Fig. 8 shows that the IIR RO is the best choice except for fast perturbations (upper row of Fig. 8) where TEAtime RO is the best option for almost all the μ/c range. The IIR controlled RO should be

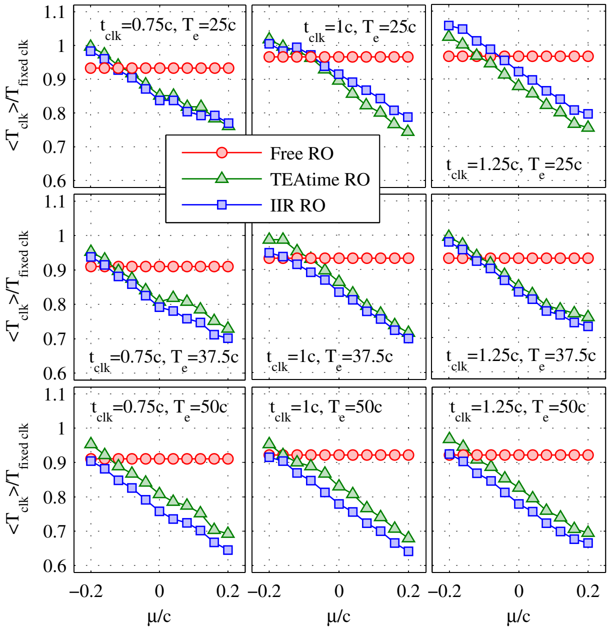


Fig. 8. Relative adaptive period for different CDN delay (t_{clk}) and variation period (T_e) values when there is a static mismatch μ between the RO and the TDC. IIR controlled RO show the best adaptation to the delay variations on all the situations. Only, for high frequency perturbations (upper row) TEAtime controlled RO is slightly better and when the mismatch is $\mu/c < -0.1$ the best option is the free running RO.

chosen, as clock generation system, to face mid-low frequency perturbations or when the clock domain is small enough to maintain the CDN small enough.

To conclude the HeDV adaptation results if we assume that the set-point $c = 64$ generates, in ideal conditions, a clock period $T_{clk} = 1$ns. Under a delay variation, due to HoDV, up to 20% and a delay variation, due to HeDV, also up to 20%; the clock period has to be set to $T_{clk} = 1.4$ns, or in the number of stages nomenclature the set-point should be changed to $c = 90$. If CDN delay and perturbation period scenario lead the adaptive clock to reduce the needed c, which assures an error free operation, up to 20%, this reduction can be translated as a reduction of 0.28ns in the clock period, which is a 70% reduction of the added safety margin.

5 Conclusions

In this paper we studied theoretically the effects of the clock distribution delay in the presence of homogeneous variations, static and dynamic. We showed that, in the presence of homogeneous dynamic variations, the clock distribution delay induces a heterogeneous variation between the clock generation circuit and the CPs on the clock domain. This induced mismatch supposes a limitation to the adaptive clock systems in terms of clock domain size.

We also argued that the heterogeneous variations may not be corrected by a concentrated adaptive clock generation system like a free running RO. To cope with heterogeneous variations we proposed a closed loop architecture with delay sensors, TDCs, disseminated along the clock domain.

Thanks to have a set-point input we propose a new clocking paradigm where the clock value is not set during the design and/or test stages. Instead of this we propose a system that tries to minimize the difference between the minimum sensors outputs and the set-point. The set-point value could be adapted as function of the timing errors during a time window and/or the performance necessities.

We modelled, at a very high level, the action of dynamic perturbations on the ring oscillator and the TDC sensors as well as the effect of a variation mismatch between them.

Since the proposed system acts like a closed loop we find some constraints for the control block when it is an IIR filter.

Finally we ran a functional simulation showing that the free running ring oscillator can not be used alone as a source of adaptive clock since its generated clock is only adapted to the variations suffered by the very near environment of the ring oscillator circuit. And that the IIR controlled ring oscillator generates the most adapted clock signal under heterogeneous variations which are likely to appear in modern ICs.

Acknowledgment. This research work has been supported by the Spanish Ministry of Science and Innovation (MICINN) and FEDER funds through project TEC2008-01856.

References

[1] Bowman, K., Duvall, S., Meindl, J.: Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. IEEE Journal of Solid-State Circuits 37(2), 183–190 (2002)

[2] Ye, Y., Gummalla, S., Wang, C.-C., Chakrabarti, C., Cao, Y.: Random variability modeling and its impact on scaled cmos circuits. J. Comput. Electron. 9, 108–113 (2010)

[3] Bowman, K.A., Duvall, S.G., Meindl, J.D.: Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. IEEE Journal of Solid State Circuits 37(2), 183–190 (2002)

[4] Steyaert, M., Van Breussegem, T., Meyvaert, H., Callemeyn, P., Wens, M.: Dc-dc converters: From discrete towards fully integrated cmos. In: 2011 Proceedings of the European Solid-State Device Research Conference (ESSDERC), pp. 59–66 (September 2011)

[5] Lee, S.-S., Kim, T.-G., Yoo, J.-T., Kim, S.-W.: Process-and-temperature compensated cmos voltage-controlled oscillator for clock generators. Electronics Letters 39, 1484–1485 (2003)

[6] Sundaresan, K., Allen, P., Ayazi, F.: Process and temperature compensation in a 7-mhz cmos clock oscillator. IEEE Journal of Solid-State Circuits 41, 433–442 (2006)

[7] Drake, A., Senger, R., Singh, H., Carpenter, G., James, N.: Dynamic measurement of critical-path timing. In: IEEE International Conference on Integrated Circuit Design and Technology and Tutorial, ICICDT 2008, pp. 249–252 (June 2008)

[8] Uht, A.: Going beyond worst-case specs with TEAtime. Computer 37, 51–56 (2004)

[9] Uht, A.: Uniprocessor performance enhancement through adaptive clock frequency control. IEEE Transactions on Computers 54, 132–140 (2005)

On-Chip NBTI and PBTI Tracking through an All-Digital Aging Monitor Architecture

Hossein Karimiyan Alidash[1], Andrea Calimera[2], Alberto Macii[2],
Enrico Macii[2], and Massimo Poncino[2]

[1] University of Kashan, Kashan, Iran
[2] Politecnico di Torino, Torino, Italy
hkarimiyan@kashanu.ac.ir,
{andrea.calimera,alberto.macii,enrico.macii,
massimo.poncino}@polito.it

Abstract. Although adaptive strategies based on the Measure-and-Control (M&C) design paradigm have been proven to be effective methods to achieve aging resilient circuits, their implementation requires accurate monitoring architectures and integrated aging sensors. This paper presents a new on-chip, fully digital monitoring architecture for tracking BTI-induced aging effects on digital ICs. The proposed solution is based on delay-to-threshold coherency of MOS devices and measures differential delay across pass-transistor chains. The aging monitor is conceived and designed as a self-contained standard gate consisting of reference and under stress sensors with embedded measurement circuitries and a control structure for data capturing. To guarantee independent measurements of both Positive- and Negative-BTI, two separate aging sensor blocks are used. Detailed SPICE simulations conducted for a low-power 40nm CMOS technology indicates the actual capability of the proposed circuit to capture BTI-induced aging.

Keywords: NBTI, PBTI, reliability sensor, sensor system design, Aging Measurement, Design for Reliability.

1 Introduction

As CMOS technology scales down, the raise of power densities and operating temperatures accelerates wear-out mechanisms in MOS transistors. Even though many aging effects do exist, e.g., Electromigration, Hot-Carrier Injection, Time Dependent Dielectric Breakdown, the introduction of new fabrication process and materials that prevent the growth of the leakage currents (e.g., high-k/metal-gate dielectrics) made Bias Temperature Instability (BTI) the first cause of unreliability for modern ICs [1].

The BTI alters the characteristics of MOS transistors and manifests as a time-dependent shift of the threshold voltage (Vth) in active devices, which, in turn, reflects in a progressive slowdown of CMOS gates. With a larger Vth, the ON current drained by the internal transistors during the switching transient phases gets smaller

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 155–165, 2013.

over time, and the logic gate itself experiences temporal, yet permanent performance degradation. Obviously, at the system level such effects are perceived as a reduction of the lifetime [1], defined as the time after which the propagation delay of the critical paths exceeds the guard-band and sampling errors appear in the internal registers and/or at the primary outputs.

The Reaction-Diffusion (R-D) theory, built upon interface trap and oxide charge generation, is used to describe the Negative-BTI (NBTI) and Positive-BTI (PBTI) aging effects. NBTI relates to PMOS transistors and refers to the Vth increase when a negative gate bias is applied between the gate and the source terminals; PBTI, occurs on NMOS transistors under a positive bias. In both cases Vth shows a partial recovery when the electrical stress is removed. It is also worth noticing that while in previous technologies PBTI was a secondary order effect, with the introduction of high-k dielectrics it cannot be longer considered as negligible [1].

Needless to say, mitigating BTI-induced aging at design time has become a must and many orthogonal approaches have been introduced in recent design flows in order to efficiently tackle the problem. At the silicon level, manufacturing process integrate customized fabrication techniques that can deliver devices which are intrinsically less sensible to the BTI [3].

At the circuit and architectural level, many methodologies have been developed, from those which make use of NBTI tolerant standard cells library [4] or that use power-gating as a natural anti-aging [6], to those that perform NBTI-aware gate sizing [5], early aging detection and mitigation [9], or that exploit other dedicated architectural solutions for pipelined structures, like those used in the Penelope processor [10]. A large class of circuit level approaches uses preventive strategies like the guard-banding, e.g., device up-sizing, scaling of the operating frequency or supply voltage [8]. Obviously, the guard-banding incurs throughput and power penalties over the entire lifetime of a design [11].

At the system level, more sophisticated, yet effective techniques rely on the M&C paradigm, under which aging-induced performance penalties are sensed and controlled by means of dedicated performance knob like adaptive body biasing (ABB) [6] or dynamic voltage and frequency scaling (DVFS) [8]. Key limitation of such M&C strategies is the availability of proper monitoring units and sensor circuits that can probe the system in real-time.

Many BTI sensors have been proposed, from those based on basic MOSFET parameters sensing [12] and propagation delay measurement through ring oscillator (RO) [14][17] [18], to those that measure the control voltage of Phase-Locked Loop (PLL) [19]. However most of them show severe limitations that can be summarized as follows. First they fail to address the time dependency of the BTI; second, do not allow individual and independent measurement of PBTI and NBTI; third, they ignore the existing relationship of BTI to signal activity; forth, the need of external and mixed-signal circuitries to sample, decode and collect aging measurements.

The contribution of this paper is to present a new on-chip all-digital aging sensor circuit that overcomes such limitations. The proposed sensor exploits the *Delay-to-Vth* relationship [1], and uses delay measurement to capture aging effect on NMOS and PMOS independently. The monitoring architecture can collect aging information across the chip and provides an effective, low cost, all-digital solution without the need for any external equipment.

The rest of paper is organized as follows. Section 2 reviews some state-of-the-art BTI sensor circuits. Section 3 presents the basic idea of the proposed method and describes the sensor block. Finally, simulation and analysis results are presented in Section 4, while Section 5 gives a brief summary and some conclusion remarks.

2 BTI Sensor Circuits

Early sensors were based on direct BTI sensing through Vth or drain current measurement [1][2]. However, monitoring small voltage drift and small currents through on-the-fly measurement techniques [12] requires current/voltage references and analog blocks which are not suitable for typical on-chip low supply voltages, and so, they require sophisticated off-chip measurement circuitries.

Indirect NBTI measurement techniques, instead, are mostly based on the fact that in the digital domain, propagation delay is an accurate and effective expression of the BTI effect. Ring-Oscillator (RO) is an example of indirect aging measurement block, where shift in the oscillating frequency is used as an indicator of the Vth shift [17][20]. More sophisticated approaches use a pair of nearly-identical ROs, where one RO is stressed, while the other one is used as reference; a measurement circuitry compares output phases and generates a ‘beat’ signal with frequency proportional to aging [14][18]. The authors of [15] adopt a similar strategy and translate the frequency measurements to the Vth shift using a calibration mechanism.

Even though ROs can be implemented with compact layouts, their deployment needs external time- or frequency-domain processing equipment. Moreover, using frequency degradation of a stressed circuit is not a true representation of the PMOS Vth shift, since the measured metrics also includes the PBTI effects of the NMOS devices, which, therefore, cannot be isolated. Another issue with the conventional RO-based structure is that there is no means to characterize the signal dependency of the BTI effects, as the duty-cycle of each transistor is fixed at 50%.

In [13] aging-induced performance degradations are used as a metric to detect timing failure of sequential circuits; the authors of the same work propose an adaptive compensation technique which is based on time borrowing [13]. The idea requires special sequential elements which impose large area and power penalties.

Other works propose the use of PLL as viable solution to capture any possible variations stemming from process, environment, or temporal NBTI, as they use the VCO control voltage inside the PLL as mirror of performance degradation [19]. A similar technique has been proposed that uses DLL [15]. Both solutions require significant area and power consumptions, and the measures they take, which are not digital, require off-chip analog-to-digital conversion.

Other proposed sensors are based on charge and discharge time of PMOS gate capacitor and its dependency on aging. The returned measures are used to control a RO in the DRM method [16].

A different class of sensor circuit exploits the regular structure of SRAM cells to detect BTI. The increase of the Vth affects the symmetry of the two cross-coupled inverters inside the cell, which, in turn, induces a shift on the voltage transfer curve of the memory [22]. This effect is deployed in a BTI sensor that is embedded in the SRAM arrays to track both NBTI and PBTI degradation [21]; failure during a Read or Write operation through the sensor indicates excessive aging. To notice, however, that these kind of sensors are fault detectors, rather than time-continuing aging sensors.

3 Proposed Aging Sensor

As described by the R-D theory, BTI shows strong dependence to circuit parameters and data pattern. In brief: 1) MOS devices age over time and the aging rate depends on the stress and recovery time ratio; 2) when the signal frequency is greater than 100Hz, the BTI degradation is frequency independent; 3) aging rate is temperature dependent, as BTI magnitude gets larger at higher on-chip temperatures [1]. Thereby capturing accurate aging profiles requires that sensors can work under a similar environmental and electrical condition as the main circuit.

The proposed monitoring approach is available for both NBTI and PBTI, thus allows applying control and tuning strategies, like body biasing, to NMOS and PMOS layout regions separately. Fig. 1 shows the structure of the NBTI aging block. It uses differential delay measurement, and consists of two NBTI sensors, the Trigger&Timing Control, and the delay measurement units (TDCs). The two NBTI sensors are identical, but one is "Under Aging Stress" and the other is kept as "Reference", i.e., idle for most of the time.

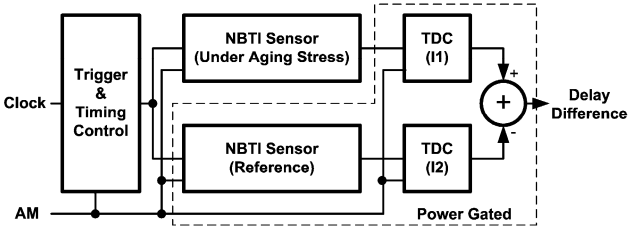


Fig. 1. The proposed NBTI measurement block

The block has two operating modes: aging (AM='1'), measurement (AM='0'). During the aging mode the Trigger&Timing Control provides a trigger signal with a specific duty cycle for both the sensors. The reference sensor, which is isolated from the trigger signal, does not age, while the other sensor ages accordingly. When a measurement is needed, the trigger pulse passes into both sensors and propagate to the other end. Since the "Under Stress" sensor is aged, its propagation delay is longer compared to the "Reference" sensor. The amount of delay difference between two sensors is a measure of the aging rate.

The timing and control circuit is used to generate trigger signal with different duty cycles. In the critical path list, those with larger duty cycle are more subject to aging.

Based on provisioned mean duty cycle in the critical path, designer can choose largest possible duty cycle value as sensor degradation rate, or connect sensor input directly to the critical path's primary output.

The core of the monitor is made up of the two BTI sensors, each one consisting of a chain of MOS devices in the pass transistor connection. Fig. 2(a) shows the implementation of the NBTI sensor, which is constructed by PMOS devices (MP1-MPn). The trigger signal is fed to one end of the chain while the other end is monitored for delay measurement. Due to the reduced swing property of PMOS transistors in the pass transistor configuration [22], NMOS devices (MN1-MNn) are added to keep the gate-to-source (Vgs) and gate-to-drain (Vgd) voltage swings in full stress range, i.e. GND to VDD. Same NMOS devices are added to the reference sensor to keep both circuits electrically identical. The PBTI sensor has the same structure and constructed by replacing PMOS device with NMOS devices.

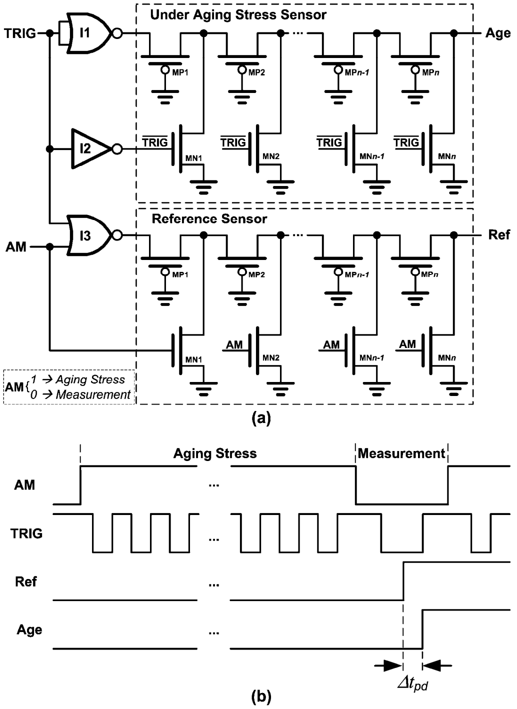


Fig. 2. (a) NBTI Sensor implementation, and (b) differential delay measurement

Using waveforms in Fig. 2(b), the NBTI sensor circuit operates as follows. During the aging mode (AM='1'), the NOR gate (I3) is blocking the TRIG signal, and all NMOS devices are on, keeping all drain and source nodes at the GND level. Meantime, the under aging section receives the TRIG input with a specific duty cycle. The gate oxide of PMOS devices faces full stress and recovery condition, same as PMOS devices located in the logic, and so ages accordingly. During measurement mode (AM='0'), the NOR gate (I3) passes the TRIG signal to the reference sensor.

Although both sensors receive same trigger signal, but due to the NBTI effect, the aged output ('Age') arrives later than the reference output ('Ref').

It has been shown that the degradation in the threshold voltage and the corresponding circuit delay have the same power-law dependency on time [1]. Therefore, by monitoring the change in the gate delay, the threshold voltage degradation can be easily estimated with a high degree of accuracy.

The propagation delay of the NBTI sensor (t_{pd}) in Fig. 2 is composed of the delay of the input gates (t_{nor}) and the delay of the pass transistors chain. According to the Elmore RC delay model [22], the propagation delay of the chain in the reference ($t_{\mathrm{tg,ref}}$) and the aged ($t_{\mathrm{tg,age}}$) sensors are estimated by the following equations:

$$\begin{aligned} t_{tg,ref} &\approx 0.69\sum_{k=0}^{n} kC_{eq,N}R_{eq,N} = 0.69C_{eq,N}R_{eq,N}\frac{n(n+1)}{2} \\ t_{tg,age} &\approx 0.69\sum_{k=0}^{n} kC_{eq,N}\left(R_{eq,N}+\Delta R_{eq,N}\right) = 0.69C_{eq,N}\left(R_{eq,N}+\Delta R_{eq,N}\right)\frac{n(n+1)}{2} \end{aligned} \tag{1}$$

where $C_{eq,N}$ and $R_{eq,N}$ are equivalent channel capacitance and channel resistance of each PMOS transistor, respectively, and n is the number of stages. Applying stress to devices located in the under aging section, they show a larger Vth compared to those devices located in the reference section. This deviation from the nominal Vth, translates to a change in the equivalent channel resistance $\Delta R_{eq,N}$ which in turn alter the propagation delay through the RC network.

Except the input NOR gates (I1 and I3), the remaining component of delay is just due to the PMOS devices. Using identical devices on each sensor and applying (1), the differential value of the propagation delay (Δt_{pd}) can be approximated by the following equation:

$$\Delta t_{pd} = t_{tg,age} - t_{tg,ref} \approx 0.69C_{eq,N}\Delta R_{eq,N}\frac{n(n+1)}{2} \tag{2}$$

According to (2) any change on device parameter, affects the differential value of the propagation delays by a multiply of '*n(n+1)/2*'. In other words, aging effect on the delay of a single PMOS device is amplified by a high gain value. This Aging Delay Amplification (ADA) actually spreads out the propagation delay and so makes it suitable for on-chip digital measurement.

On-chip delay measurement requires time-to-digital converter (TDC) circuit [22]. Fig. 3 shows the internal design of delay measurement circuit in the BTI sensor. It consists of a tapped linear delay line, and flip-flops (FFs) connected to each tap.

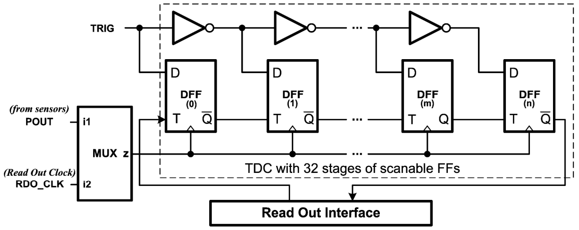


Fig. 3. The delay measurement setup using linear delay line with locally scanable FFs

In the measurement session and during the capture phase, FFs are in normal mode to sample the TRIG signal using sensor output (OUT) as the reference clock. In those taps that delay is larger than setup time, FFs capture data properly while others remain intact. The number of FFs with successful captured data is a measure of the delay value. The captured data is a binary word proportional to the delay introduced by the sensor. During the read-out phase of measurement, FFs are in the test mode and synchronized with the RDO_CLK signal enabling main controller to read back the captured aging information.

4 Simulation and Analysis Results

The proposed aging sensor has been designed and simulated using a high-performance 45nm CMOS technology. The simulation results were obtained at room temperature with 1.1V nominal supply voltage using SPICE simulator. The MOS devices in the chains are mapped to minimum feature size and are one-by-one similar, thus to make propagation delay of sensors almost the same. Fig. 4 depicts the simulation result of the NBTI sensor.

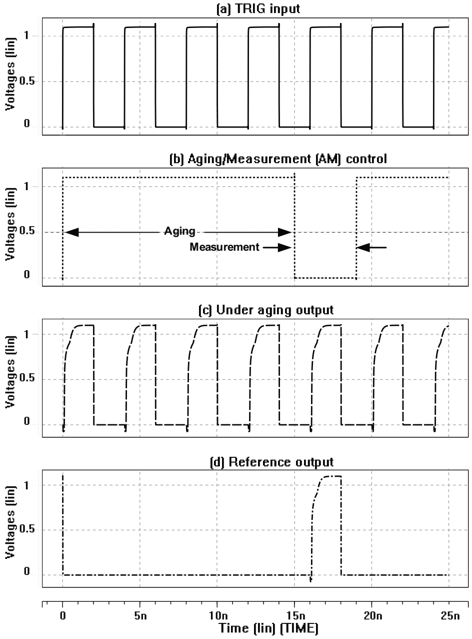


Fig. 4. Simulation result of the NBTI sensor, (a) TRIG pulse, (b) A/M control, (c) under aging sensor, and (d) reference sensor output

During the aging mode, the TRIG signal is blocked in the reference sensor, while the other sensor receives it for a long time and so ages accordingly. During the measurement mode the TRIG signal is delivered to both sensors which then follow the capture and read-out phase in the TDCs. The PBTI sensor measures the aging effect in the NMOS devices and has similar waveforms and operation phases.

Table 1 shows the propagation delay and its deviation in the NBTI and PBTI sensors with varying number of chain stages. It also shows delay changes with respect to 5 and 10 percent increase of the Vth in each aged MOS transistor. The Vth shift is used to emulate the BTI aging effect. Using the reported delay values in this table and with the aim of easy and low-cost delay measurement, number of stages in both sensors is set to 16. Fig. 5 shows the propagation delay change in the NBTI and PBTI sensors with increasing value of the Vth in each MOS transistor. This graph shows that propagation delay is changing in an approximately linear rate as the Vth changes. The simulation results of Table 1 and Fig. 5 indicate that the ADA effect of the chain manifests in large delay change at the sensor output.

Table 1. Propagation delay of sensors with different stage numbers and its change after 5% and 10% change in the Vth

Type	Stages	t_{pd} (ps)	Δt_{pd} (ps) @ %5 ΔVth	Δt_{pd} (ps) @ %10 ΔVth
NBTI Sensor	2	18.37	0.35	0.99
	4	38.35	2.74	4.61
	8	115.0	9.49	19.14
	16	407.8	38.48	75.05
PBTI Sensor	2	7.14	0.27	0.54
	4	17.66	0.94	1.77
	8	55.12	4.8	7.75
	16	190.06	13.6	27.6

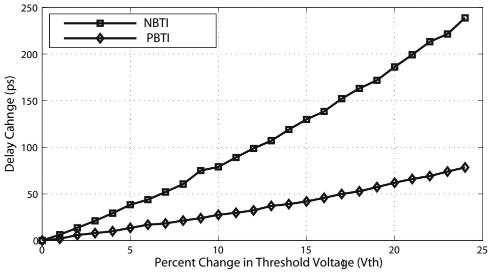


Fig. 5. The propagation delay change of the NBTI and PBTI sensors w.r.t Vth shift

This amplified delay sensitivity to Vth is synonym of larger dynamic range, which in turn, allows easier and less expensive measurement and digital conversions. Considering that the a minimum sized inverter in the target technology under a Fan-out of 4 gives a delay of 10ps, the dynamic range of the propagation delay in Fig. 5 guarantees enough resolution; thereby on-chip aging sensor through direct delay measurement using a TDC with 25 stages is deployed [23].

The TDC operation is modeled using HDLs, and simulated along with delay data delivered from SPICE level simulation. Fig. 6 shows the simulation results, indicating the differential value of the propagation delay reached to the TDC. Despite the quantization effect, comparing the SPICE level simulation result of Fig. 5 with digitized values in Fig.

6, one can observe similar trend. This underlines the circuit's capability to capture the aging effect in digital form without using off-chip measurement equipment.

As final remark, Table 2 provides a qualitative analysis of the proposed solution w.r.t. existing aging sensor. The metrics used for the comparison include the sensing method, namely, the variable sensed to extract aging profiles, how and where additional circuitries are placed, the measured effect and the design objective. Most published circuits are limited to NBTI and intended to process characterization rather than providing feedback for M&C. Those methods based on fault detection are just for end-of-life detection, not suitable for M&C strategies. Other Vth or Drain current measurement based methods need analog blocks which are not feasible in low supply voltages, and generally require off-chip equipment. In contrast, the proposed sensor is targeted for circuit compensation, it measures both NBTI and PBTI aging and makes the resulted digital signature available on-chip for any possible circuit enhancement in the run-time. Also, the digital interface enhances testability and programmability of the sensor.

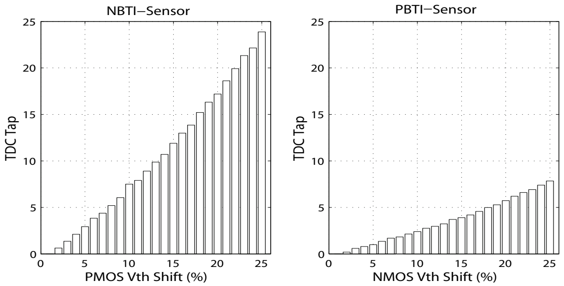


Fig. 6. Quantized differential delay value as it captured on the TDC's FFs in NBTI and PBTI sensor

Table 2. Several aging sensor design comparison

Sensor	Sensed Variable	On-Chip	NBTI/PBTI	Objective
[13]	Setup Time	-	Mixed	Binary Fail Detect
[14]	Frequency	No	NBTI/PBTI	Circuit Tuning
[15]	Vth	No	NBTI	Characterization
[17]	Frequency	No	NBTI	Characterization
[18]	Delay	Yes	NBTI	Circuit Tuning
[20]	Frequency	No	NBTI/PBTI	Characterization
[21]	Delay	No	NBTI/PBTI	Binary Fail Detect
This work	Delay	Yes	NBTI/PBTI	Circuit Tuning

5 Conclusion

In this work, an on-chip all-digital aging sensor circuit suitable for adaptive mitigation methods was proposed. The proposed sensing method was aimed to capture the NBTI and PBTI aging information of the PMOS and NMOS devices, individually. The basic concept, its practical realization, and sensitivity to aging have been described. The sensor is suggested to be embedded through the core, and exposed to the same environmental and electrical conditions to faces the same aging rate as the core logic. Using the on-chip digitizer deployed in the sensor block, aging information was provided to processing unit in digital form without requiring any analog block or off-chip measurement equipment. The feasibility and accuracy of the test structure is demonstrated in a 40nm CMOS technology. The results indicate that the sensor is able to measure aging rate of PMOS and NMOS device individually with minimal time and complexity, and thus enable run-time optimization and yield improvement.

References

1. Paul, B.C., Kang, K., Kufluoglu, H., Alam, M.A., Roy, K.: Impact of NBTI on the temporal performance degradation of digital circuits. IEEE Electron. Device Letters 26(8), 560–562 (2005)
2. Bhardwaj, S., Wang, W., Vattikonda, R., Cao, Y., Vrudhula, S.: Predictive Modeling of the NBTI Effect for Reliable Design. In: IEEE CICC 2006, pp. 189–192 (2006)
3. Scarpa, A., Ward, D., Dubois, J., van Marwijk, L., Gausepohl, S., Campos, R., Sim, K.Y., Cacciato, A., Kho, R., Bolt, M.: Negative-bias temperature instability cure by process optimization. IEEE Tran. on Electron. Devices 53(6), 1331–1339 (2006)
4. Basu, S., Vemuri, R.: Process Variation and NBTI Tolerant Standard Cells to Improve Parametric Yield and Lifetime of ICs. In: Proc. of the IEEE ISVLSI 2007, pp. 291–298 (2007)
5. Yang, X., Saluja, K.: Combating NBTI Degradation via Gate Sizing. In: Proc. of the 8th Inte.l Symp. on Quality Electronic Design (ISQED 2007), pp. 47–52. IEEE Computer Soc. (2007)
6. Calimera, A., Macii, E., Poncino, M.: NBTI-Aware Clustered Power Gating. ACM Trans. Des. Autom. Electron. Syst. 16(1), Article 3 (2010)
7. Qi, Z., Stan, M.R.: NBTI resilient circuits using adaptive body biasing. In: Proc. of the 18th ACM Great Lakes Symposium on VLSI (GLSVLSI 2008), pp. 285–290 (2008)
8. Basoglu, M., Orshansky, M., Erez, M.: NBTI-aware DVFS: a new approach to saving energy and increasing processor lifetime. In: Proc. of the ISLPED 2010, pp. 253–258 (2010)
9. Dadgour, H., Banerjee, K.: Aging-resilient design of pipelined architectures using novel detection and correction circuits. In: Proceedings of the DATE 2010, pp. 244–249 (2010)
10. Abella, J., Vera, X., Gonzalez, A.: Penelope: The NBTI-Aware Processor. In: Proceedings of the 40th Annual IEEE/ACM MICRO 40, pp. 85–96. IEEE Computer Society (2007)
11. Chan, T., Sartori, J., Gupta, P., Kumar, R.: On the efficacy of NBTI mitigation techniques. In: Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1–6 (2011)
12. Denais, M., Parthasarathy, C., Ribes, G., Rey-Tauriac, Y., Revil, N., Bravaix, A., Huard, V., Perrier, F.: On-the-fly characterization of NBTI in ultra-thin gate oxide PMOSFET's. In: IEEE International Electron Devices Meeting. IEDM Technical Digest, pp. 109–112 (2004)

13. Dadgour, H.F., Banerjee, K.: A built-in aging detection and compensation technique for improving reliability of nanoscale CMOS designs. In: 2010 IEEE IRPS, pp. 822–825 (2010)
14. Keane, J., Wang, X., Persaud, D., Kim, C.H.: An All-In-One Silicon Odometer for Separately Monitoring HCI, BTI, and TDDB. IEEE JSSC 45(4), 817–829 (2010)
15. Keane, J., Kim, T.-H., Kim, C.H.: An on-chip NBTI sensor for measuring PMOS threshold voltage degradation. In: Proc. of the ISLPED 2007, pp. 189–194 (2007)
16. Singh, P., Karl, E., Sylvester, D., Blaauw, D.: Dynamic NBTI management using a 45nm multi-degradation sensor. In: 2010 IEEE CICC, pp. 1–4 (2010)
17. Karl, E., Singh, P., Blaauw, D., Sylvester, D.: Compact In-Situ Sensors for Monitoring Negative-Bias-Temperature-Instability Effect and Oxide Degradation. In: IEEE International Solid-State Circuits Conference, ISSCC 2008, pp. 410–623 (2008)
18. Kim, K.K., Wang, W., Choi, K.: On-Chip Aging Sensor Circuits for Reliable Nanometer MOSFET Digital Circuits. IEEE Transactions on Circuits and Systems II: Express Briefs 57(10), 798–802 (2010)
19. Kang, K., Park, S.P., Kim, K., Roy, K.: On-Chip Variability Sensor Using Phase-Locked Loop for Detecting and Correcting Parametric Timing Failures. IEEE Trans. on Very Large Scale Integration (VLSI) Systems 18(2), 270–280 (2010)
20. Kim, J.-J., Linder, B.P., Rao, R.M., Kim, T.-H., Lu, P.-F., Jenkins, K.A., Kim, C.H., Bansal, A., Mukhopadhyay, S., Chuang, C.-T.: Reliability monitoring ring oscillator structures for isolated/combined NBTI and PBTI measurement in high-k metal gate technologies. In: IEEE International Reliability Physics Symposium (IRPS), pp. 2B.4.1–2B.4.4 (2011)
21. Qi, Z., Wang, J., Cabe, A., Wooters, S., Blalock, T., Calhoun, B., Stan, M.: SRAM-based NBTI/PBTI sensor system design. In: 47th ACM/IEEE Design Automation Conference (DAC), pp. 849–852 (2010)
22. Weste, N., Harris, D.: CMOS VLSI Design: A Circuits and Systems Perspective, 4th edn. Addison-Wesley Publishing Company (2010)
23. Henzler, S.: Time-To-Digital Converters, 1st edn. Springer Publishing Company, Incorporated (2010)

Two-Phase MOBILE Interconnection Schemes for Ultra-Grain Pipeline Applications

Juan Núñez, María J. Avedillo, and José M. Quintana

Instituto de Microelectrónica de Sevilla (IMSE-CNM-CSIC) & Universidad de Sevilla
Av. Américo Vespucio s/n, 41092, Sevilla, Spain
{jnunez,avedillo,josem}@imse-cnm.csic.es

Abstract. Monostable to Bistable (MOBILE) gates are very suitable for the implementation of gate-level pipelines which can be achieved without resorting to memory elements. MOBILE operating principle is implemented using two series connected Negative Differential Resistance (NDR) devices with a clocked bias. This paper describes and experimentally validates a two-phase clock scheme for such MOBILE based ultra-grain pipelines. Up to our knowledge it is the first MOBILE working circuit reported with this interconnection architecture. The proposed interconnection architecture is applied to the design of a 4-bit Carry Look-ahead Adder.

Keywords: Negative Differential Resistance (NDR), Nanopipeline, Monostable to Bistable Logic Elements (MOBILE).

1 Introduction

Different emerging devices like Resonant Tunneling Diodes (RTDs), tunnel transistors or molecular RTD devices exhibit Negative Differential Resistance (NDR) in their *I-V* characteristic. Many circuits taking advantage of it have been reported covering different applications and with different goals including high speed, low power or reduced device count [1], [2], [3] so that design techniques exploiting this feature at different levels (circuit, architecture, ..) are currently an area of active research.

From the design point of view, the NDR characteristics are very attractive. On one hand, it can be exploited in non-linear circuits like oscillators or frequency dividers. On the other, it is useful in the implementation of memories due to the existence of stable states associated to the inclusion of NDR elements. In particular, the Goto pair [4] is well known. The circuit consists of two NDR devices connected in series leading to three operating points, two stable and one unstable. The two stable points can be used to represent and store data.

On the basis of the Goto pair, logic circuits which operation is based on a Monostable to a Bistable transition (MOBILE) have been developed. MOBILE gates are implemented operating two series connected NDR devices with a switching bias. There are two interesting characteristics of MOBILE gates in comparison to conventional logic gates implementations.

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 166–174, 2013.

First, they increase the functionality implemented by a single gate in comparison to MOS and bipolar technologies thus reducing circuit complexity. In particular, the operating principle of MOBILE is extremely well suited to implement the arithmetic operation on which Threshold Gates (TGs) are based [5]. Different topologies for RTD TGs and Multi-Threshold Threshold gates have been reported and experimentally validated.

Second, the latching property of MOBILEs arising from their NDR characteristic allows the implementation of gate-level pipelines which can be achieved without resorting to memory elements [1], [6] , and which do not exhibit the functional limitation of conventional CMOS solutions based on dynamic logic, allowing only non-inverting blocks to be chained.

Originally, it was proposed to operate MOBILE gates in a gate level pipelined fashion using a four-phase clock scheme. However, operating frequency (or throughput) depends not only on the number of gate levels, but also on the number of clock-phases, since clock period needs to accommodate all the phases. Thus, from the point of view of speed a two-phase scheme is very attractive.

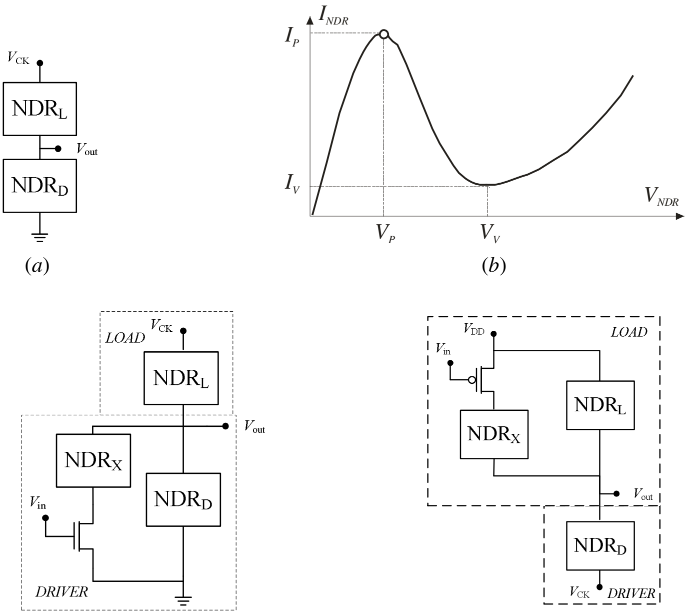
This paper explores in depth and experimentally validates a two-phase clock scheme for MOBILE based fine-grain pipelines.

The paper is organized as follows: in Section 2, MOBILE logic style is described. In Section 3, two-phase gate-level MOBILE pipelines are introduced, showing experimental results that validate their operation. In Section 4 a Carry Lookahead Adder which has been designed using the proposed pipeline is described. Finally, some key conclusions are given in Section 5.

2 Background

2.1 MOBILE Logic Gates

The MOBILE [5] in Fig. 1a is an edge-triggered current controlled gate which consists of two devices exhibiting NDR in their *I-V* characteristic (Fig. 1b), connected in series and driven by a switching bias voltage, V_{CK}. When V_{CK} is low, both NDRs are in the on-state and the circuit is monostable. Increasing V_{CK} to an appropriate maximum value ensures that only the device with the lowest peak current switches from the on-state to the off-state. Output is high if the driver NDR switches and it is low if the load does. Logic functionality can be achieved if the peak current of one of the NDR devices is controlled by an input. In the configuration of the rising edge-triggered inverter MOBILE shown in Fig. 1c, the peak current of the driver NDR can be modulated using the external input signal V_{in}. Transistor behaves like a switch, so that for a low input, current flows only through NDR_D, but for a high input, the effective peak current of the driver is the sum of the peak currents of NDR_D and NDR_X. Replacing the single transistor in Fig. 1c by an NMOS transistor network, other logic functions are implemented. NDR peak currents are selected such that the value of the output depends on whether the network transistor evaluates to “1” or to “0”. Figure 1d depicts a falling edge triggered inverter. Note that branch implementing functionality is now in parallel to the load NDR and uses a p-type transistor.



(*a*) (*b*)

(*c*) (*d*)

Fig. 1. MOBILE circuits. (*a*) NDR *I-V* characteristic. (*b*) Basic MOBILE. (*c*) Rising edge-triggered MOBILE inverter. (*d*) Falling edge-triggered MOBILE inverter.

A sufficiently slow V_{CK} rising (or falling) is required for MOBILE operation [7]. That is, there is a critical rise time for the switching bias below which the gate does not operate correctly. Under that critical rise time, there is at least one input combination for which the gate does not produce the expected logic output. Since AC currents associated to internal parasitics and output capacitive loads (fan-out) are more important for faster clock changes, the ideal MOBILE operating principle, based on peak currents comparison, can be substantially modified. This critical value depends on both circuit (NDR peak currents, fan-out …) and technological parameters. That is, design requires taking into account these AC currents in order to guarantee the desired relationship between load and driver currents for each input combination when V_{CK} approaches $2V_p$, being V_p the peak voltage of the NDRs (see Fig. 1a).

Rising (falling) edge-triggered MOBILE logic gates evaluate the inputs with the rising (falling) edge of the bias voltage and hold the logic level of the output while the bias voltage is high (low), even though the inputs change (self-latching operation [8]). The output returns to zero (to one) with the falling (rising) edge of the clock until the next evaluation. The self-latching operation allows the implementation of gate-level pipelined architectures without extra memory elements [1] and without the functional limitations of dynamic based solutions like the widely used domino logic style.

2.2 Interconnecting MOBILE Gates

As it was stated in previous section, and assuming rising-edge MOBILEs, there are four steps in the operation of each gate: evaluation (clock rises), hold (clock high), reset (clock falls) and wait (clock low). Gate-level pipelining is possible if each MOBILE gate evaluates while those driving it are in the hold state. In this way, it is guaranteed that inputs to each gate are stable during evaluation, and that the reset of the MOBILE gates does not affect those they drive, since they have already evaluated when it happens. Thus, memory elements are not required. Note that this is true both for inverting and non inverting MOBILE stages, and even when adding an output stage (static inverter or buffer) to the output of the MOBILE blocks to ease management of fan-out and interconnections. That is, fulfilling above stated constraint allows ultra fine-grain pipeline operation where both inverting and non inverting stages are allowed. In domino solutions only non inverting stages are possible which complicates logic design (inverters need to be pushed towards the inputs or some parts of the circuits are duplicated), unless a double rail implementation is used which almost duplicates device count.

Conventionally, and because of the four steps in MOBILE operation, cascaded rising edge-triggered MOBILE gates are operated in a pipelined fashion using a four-phase overlapping clocking scheme shown in Fig. 2a [1]. $V_{CK,\,i}$ is delayed with respect to $V_{CK,\,i-1}$ by $T/4$, being T the clock period. In this way the i^{th} stage evaluates (rising edge of $V_{CK,\,i}$) while the $(i-1)^{th}$ stage is in the hold phase ($V_{CK,\,i-1}$ high). Four clock signals are enough, since the first phase can be used for the fifth level and so on. In previous section, it was stated that there is a critical rise time below which the gate does not operate correctly.

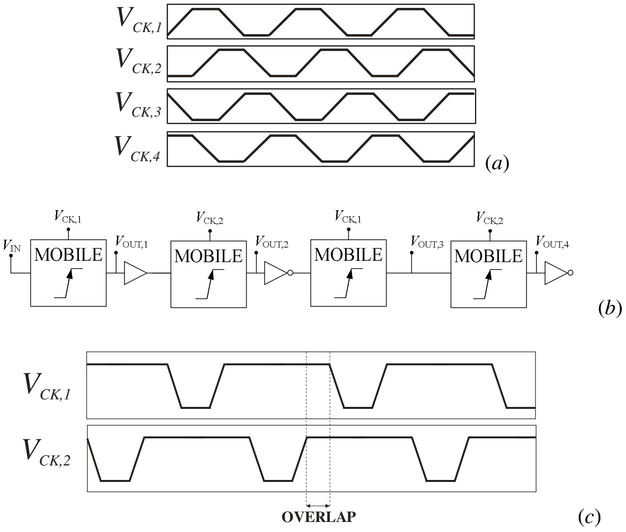


Fig. 2. (a) Four-phase clock scheme. (*b*)-(*c*) Block diagram and clock waveforms of the two-phase chain of inverters.

This explains the clock shape with equal rise, high, fall and low times. Thus, for this scheme four gates/stages serially evaluate in one clock period.

However other schemes are compatible with the constraint that one stage evaluates while preceding stages are in hold state. Moreover, the constraint can be somewhat relaxed making possible other simpler schemes.

Single phase scheme has been proposed [9]. However there are two drawbacks associated to the single-phase solutions. First, negative edge-triggered MOBILE are used which requires p-type transistors. This translates in larger transistors and so in larger parasitic capacitances which degrade gate speed. Second, they exhibit limited clock-skew tolerance. Recently, we have proposed a two-phase scheme [10] which overcomes both issues while being similar in terms of throughput and latency.

Next section describes the two-phase interconnection scheme and shows experimental results of fabricated circuits validating the approach.

3 Two-Phase MOBILE Networks

An alternative solution consists of the design of networks of only positive edge triggered MOBILE gates operated with an overlapping two phase clock scheme as shown in Figures. 2b and 2c. It can be easily seen that each gate evaluates while preceding one is in the hold state, and that only two stages serially evaluates in one clock period. Note that inter-gate elements (inverting or not inverting) can also be added if required by logical (to increase design flexibility), or electrical (for example, efficient handling of large loads) considerations.

It is interesting to make some comments concerning the amount by which the clock-phases overlap. Due to the edge-triggered nature of MOBILE evaluations, required minimum overlap is generally small, especially when inverters/buffers are used between MOBILE blocks, since, as it was anticipated, the interconnection constraint can be relaxed. What is required for proper operation is that current stage takes a decision before it sees the reset of the previous stage. That is, before the output of the preceding MOBILE blocks reaches a low level output voltage and it propagates through the inter-MOBILEs elements. The maximum overlap is only limited by the maximum allowable duty cycle of the clock, which is determined by the minimum time required for the reset of the MOBILE gates. MOBILE output must discharge to zero before an evaluation. Thus, overlap is fixed such that clock skew is tolerated.

Two-phase clocked chains of MOBILE inverters have been designed and fabricated. These structures have been implemented with MOS-NDR devices (circuits made up of transistors that emulate the NDR *I-V* characteristic) and the MOBILE gate topology from [11] in a 1.2V/90nm CMOS technology. Figures 3a and 3b depict the schematics of the MOS-NDR device which has been used and the schematic of a MOBILE inverter implemented with them. The design of MOBILE blocks operating at high frequencies is not straightforward. As it was previously mentioned, it is necessary to take into account AC currents associated to parasitics which can be large at high frequencies. Thus, design validation requires both an accurate modeling of layout parasitics and experimental validation.

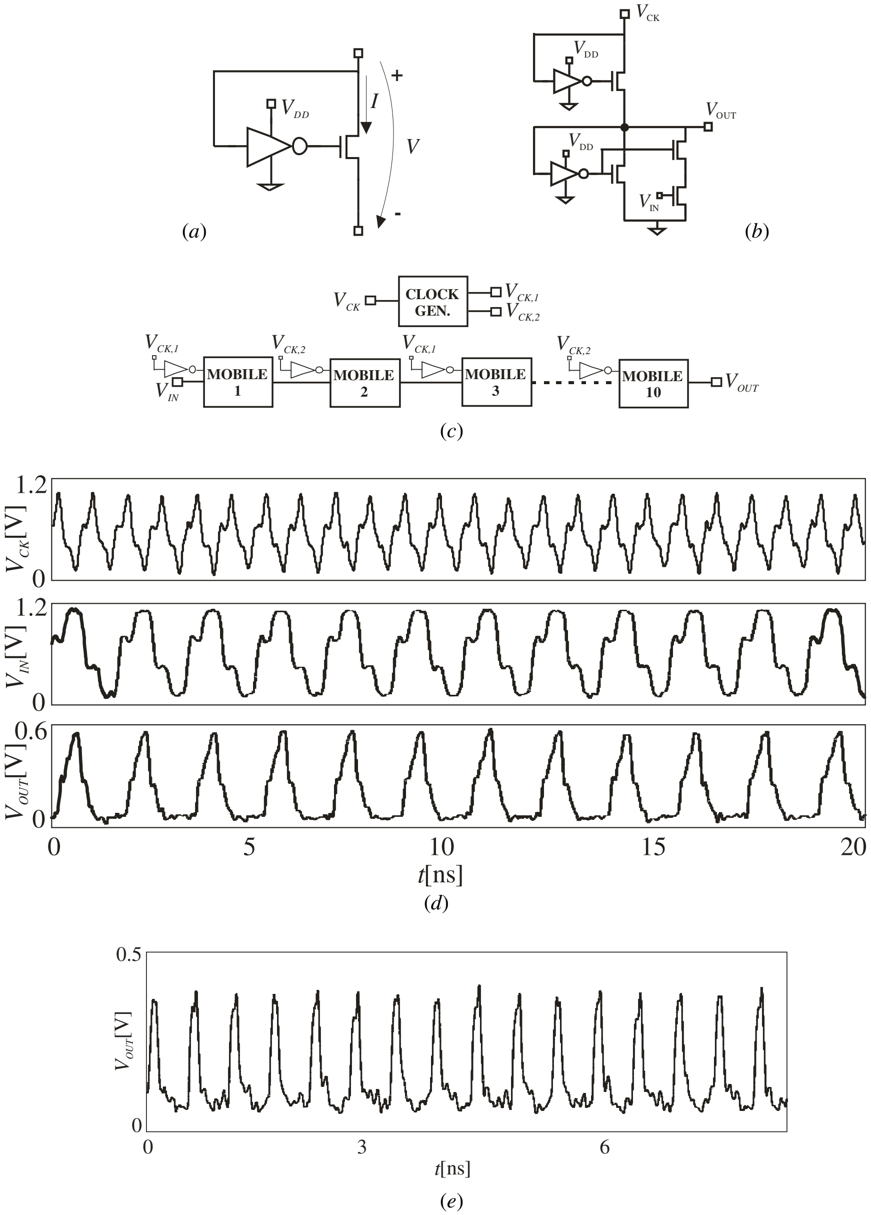


Fig. 3. (a) MOS-NDR device. (*b*) MOBILE inverter implemented with MOS-NDR devices. (*c*) Block diagram of a fabricated two-phase chain of inverters. (*d*) Measured waveforms. (*e*) Output of a fabricated two-phase chain of inverter with static buffers as interconnection stages.

Figure 3c depicts the block diagram of one of the fabricated circuits. Each MOBILE stage is an inverter similar to the one depicted in Fig. 3b. As shown in Fig. 3c, a two-phase clock generator has been also included. It provides two non-overlapped clock signals ($V_{CK,1}$ and $V_{CK,2}$) with the same frequency of the master clock (V_{CK}). Note that power clocks are avoided since the clock signal of each MOBILE circuit is applied to the input of a static inverter. The output of this inverter is used as the clock of the MOBILE inverter. In this way, the two required overlapped clocks are generated and the constraints on clock rising time are relaxed.

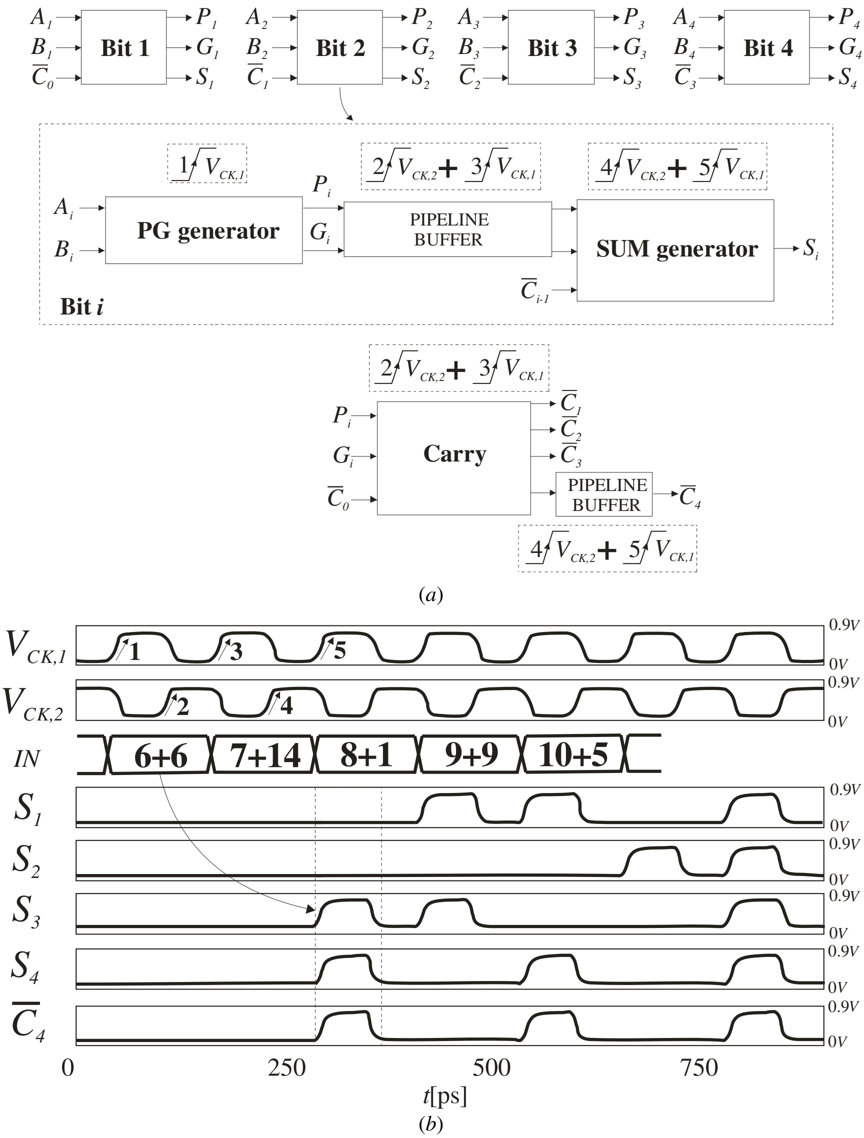
The packaged circuit has been probed and correct operation has been shown. Figure 3d depicts experimental results when a sequence alternating 0's and 1's is applied to the ten-stage pipeline. Waveforms of the master clock, the input (V_{IN}) and the output (V_{OUT}), which have been captured using the oscilloscope Agilent DSO6104A, are shown. Note that, in addition to package, there are input buffers (for V_{IN} and V_{CK}), output buffers and pads which are not shown in Fig. 3. V_{CK} and V_{IN} are 1.2V pulse trains at 1GHz and 500MHz, respectively. As expected, V_{OUT} is a periodical signal of the same frequency of the input. The 0101…01 sequence is obtained at the output of the pipeline with a latency of five clock cycles, since data is evaluated twice each cycle of V_{CK}. Note the different shapes of V_{IN} and V_{OUT} which is due to the return to zero behavior of MOBILE. Results are shown at 1GHz so that signals are attenuated by the experimental set-up. Finally, Fig. 3e shows the output of another two-phase chain of inverters incorporating static buffers between MOBILE stages. For this circuit, both V_{CK} and V_{IN} have been generated on- chip so that the input frequency is half that of the clock (1.7GHz).

4 4-bit CLA

The two-phase clock scheme has been applied in the design of a 4-bit RTD-CMOS MOBILE Carry Lookahead Adder (CLA) as a case study.

Figure 4a shows the block diagram of a 4-bit CLA. The PGs blocks propagate ($P_i = A_i$ XOR B_i), generate ($g_i = A_i$ *AND* B_i) and sum ($S_i = A_i$ XOR B_i XOR C_i) bits. The Carry block generates the carry signals C_1, C_2, C_3 and C_{out} ($c_{i+1} = G_i$ *OR(* P_i *AND* C_i*)*. The carry block is implemented with two pipeline stages using NOR gates. Implementation of XOR gates takes advantage of the possibility of using inverting and non inverting inter-stages elements so that variables and their complements are available. There are five stages in the design and so latency is two-cycles and a half. In Fig. 4a it has been marked which clock cycles corresponds to each stage.

The study uses PTM 32nm transistor model. The RTD has been modeled using a voltage-dependent current source and a capacitor in parallel and technology parameters from an experimentally validated Si-Ge RTD [8] with peak current density j_p=218KA/cm^2 and capacitor C=6fF/µm^2. Transistor lengths have been set to the minimum value associated to the technology whereas their widths are large enough to allow their operation as switches. RTD areas have selected to work at a frequency of 0.12/*FO-4* (*FO-4* is the FanOut-4 inverter delay of the technology, 14.69ps) at a supply voltage of 0.9V. Parasitics capacitances have been added to model drain and source diffusion parasitic (0.25fF), RTD-transistor contacts (0.25fF) and interconnections (1fF).



(*a*)

(*b*)

Fig. 4. (a) Block diagram of a 4-bit CLA. (*b*) Simulation waveforms of the proposed CLA showing a correct operation.

Figure 4b shows simulation waveforms of the designed CLA for selected values of the inputs (A+B), where correct operation is observed. Note that both the sum bits and the carry are obtained after the third rising edge of $V_{CK,1}$ (two-cycles and a half from the evaluation of the inputs).

5 Conclusions

The operation of two-phase gate-level pipelines based on MOBILE operating principle has been experimentally validated. Up to our knowledge it is the first time a working two-phase MOBILE network is reported. This interconnection scheme has advantages over other previously reported clock schemes for MOBILE logic networks. It is simpler than the conventional four-phase solution and leads to higher operating frequencies since only two-stage, instead of four, sequentially evaluates in one clock cycle. Unlike one phase scheme, p-type transistors are avoided and clock skew tolerance increases. The later is due to the self-latching property of MOBILE gates which makes possible an operation taking advantage of the overlapping of the two clock phases to tolerate clock skew and, in addition, avoids limitations of conventional CMOS counterparts. The design of a two-phase 4-bit RTD-CMOS CLA has been carried out as an application example.

Acknowledgement. This work has been funded by Ministerio de Economía y Competitividad del Gobierno de España with support from ERDF under Project TEC2010-18937.

References

[1] Mazumder, P., Kulkarni, S., Bhattacharya, M., Sun, J.P., Haddad, G.I.: Digital circuit applications of resonant tunneling devices. Proc. IEEE 86, 664–686 (1998)

[2] Choi, S., Jeong, Y., Lee, J., Yang, K.: A Novel High-Speed Multiplexing IC Based on Resonant Tunneling Diodes. IEEE Trans. on Nanotechnology 8(4), 482–486 (2009)

[3] Karda, K., Brockman, J., Sutar, S., Seabaugh, A., Nahas, J.: Bistable-Body Tunnel SRAM. IEEE Trans. Nanotechnology PP(9), 1 (2010)

[4] Goto, E., Murata, K., Nakazawa, K., Nakagawa, K., Moto-Oka, T., Ishibashi, Y., Soma, T., Wada, E.: Esaki diode high-speed logical circuits. IRE Trans. Elect. Comp. EC-9, 25–29 (1960)

[5] AKeyoshi, T., et al.: Weighted sum threshold logic operation of MOBILE's (Monostable Bistable Logic Element) using resonant-tunnelling transistors. IEEE Electron. Device Lett. 14, 475–477 (1993)

[6] Mohan, S., et al.: Ultrafast pipelined arithmetic using quantum electronic devices. In: IEE Proceedings of the Computers and Digital Techniques, vol. 141(2), pp. 104–110 (March 1994)

[7] Matsuzaki, H., Fukuyama, H., Enoki, T.: Analysis of Transient Response and Operating Speed of MOBILE. IEEE Trans. on Electron Devices 51(4), 616–622 (2004)

[8] Avedillo, M.J., Quintana, J.M., Pettenghi, H.: Self-Latching Operation of MOBILE Circuits using Series-Connection of RTDs and Transistors. IEEE Trans. on Circuits and Systems-II 53(5), 334–338 (2006)

[9] Núñez, J., Avedillo, M.J., Quintana, J.M.: Simplified single-phase clock scheme for MOBILE networks. Electronics Letters 47(11), 648–650 (2011)

[10] Núñez, J., Avedillo, M.J., Quintana, J.M.: Domino Inspired MOBILE Networks. Electronics Letters 48(5) (February 2012)

[11] Núñez, J., Avedillo, M.J., Quintana, J.M.: Efficient Realisation of MOS-NDR Threshold Logic Gates. IET Electronics Letters 45(23), 1158–1160 (2009)

[12] Chung, S.Y., et al.: Si/SiGe resonant interband tunnel diode with fr0 20.2GHz and peak current density 218 kA/cm2 for K-band mixed-signal applications. IEEE Electro. Device Lett. 27, 364–367 (2006)

Design of a 150 mV Supply, 2 MIPS, 90nm CMOS, Ultra-Low-Power Microprocessor

Pieter Weckx[12], Nele Reynders[1], Ilse de Moffarts[1], and Wim Dehaene[12]

[1] Katholieke Universiteit Leuven, ESAT-MICAS, 3001 Leuven, Belgium
[2] imec vzw. Kapeldreef 75, 3001, Leuven, Belgium

Abstract. This work presents a simulated ultra-low-power 16-bit sub-threshold microprocessor designed in a 90 nm CMOS technology. Transmission gate logic extended with transistor stacking is used for its high robustness to inter- and intra-die variations, while combining low power and small area. In a first implementation, the sub-threshold microprocessor has a throughput of 1 MIPS at a 4 MHz clock and a 150 mV supply with an energy per instruction of 0.74 pJ. Improved results are obtained using pipelining, which allows the microprocessor to achieve a maximum performance of 2 MIPS, an energy consumption of 0.48 pJ per instruction, an EDP of 0.23 pJ×μs and a 0.9 μW power consumption.

1 Introduction

Research for low-power circuits has become important due to the increasing demand for battery-supplied systems. These systems, e.g. wireless sensor networks and medical applications, require a prolonged autonomy using only a limited amount of stored energy. Very low energy consumption has successfully been achieved by aggressively lowering the supply voltage in order to operate at the minimum energy point [1,2,3]. This however occurs in sub-threshold operation [4] which results in increased sensitivity to inter- and intra-die variations caused by the exponential relation between the threshold voltage V_{t} and the transistor current I_{sub}. Moreover, MHz-performance is hard to reach due the the exponentially decreased drive current making sub-threshold circuits fall short of fulfilling the needs of modern ultra-low-power micro-architectures. Other approaches address these issues by operating the circuit in near- or super-threshold operation conditions combined with various performance improving techniques like architecture-level parallelism [5,6] and variability control techniques like V_t-configuration and balancing [2,6], body-biasing [2] and finger-structured parallel transistor cells [6]. Circuit robustness while still reaching acceptable performance is nonetheless possible by a careful exploration of logic families to suit sub-threshold design. This paper handles the design and simulation of a fully functional 16-bit microprocessor in a 90 nm CMOS technology node operating at a clock frequency of 4 MHz and a supply voltage of 150 mV. It is possible to reach this high MHz-performance in a variation-resilient sub-threshold design using transmission gate (TG) logic instead of static CMOS logic. A full processor is designed and simulated using differential TG logic extended with nMOS stacking to show the

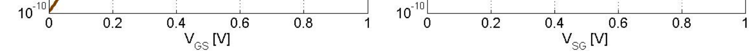
J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 175–184, 2013.

energy efficiency and potential for low-power circuits. Section 2 describes how the sub-threshold TG logic is used in the design. Section 3 presents the processor architecture. Section 4 shows simulation results. Finally, the conclusion is presented in Section 5.

2 Sub-threshold Logic Design

2.1 Variability and Leakage

Fig. 1 shows the transistor's drain current as function of the gate voltage for all process corners of the used 90 nm CMOS technology. At a supply of 150 mV, variation is exponentially larger as compared to the nominal voltage of 1 V. The spread for pMOS transistors is larger than nMOS transistors because the former are further in the sub-threshold region due to their higher threshold voltage. Additionally, due to the sub-threshold operation, the *on* and *off* transistor currents are both sub-threshold currents resulting in reduced $I_{\mathrm{on}}/I_{\mathrm{off}}$-ratios. The leakage current I_{off} is only a few orders of magnitude smaller than the drain current I_{on}. Moreover, the poor ratio of pMOS on-current versus nMOS off-current ($I_{\mathrm{on,p}}/I_{\mathrm{off,n}}$) results in poor noise margins for logic gates. Table 1 summarizes the drain current characteristics for a supply voltage V_{DD} of 150 mV and 1 V.



(a) nMOS transistor. (b) pMOS transistor.

Fig. 1. I_{ds} as function of V_{gs} on a logarithmic scale in all process corners

2.2 Combinational Logic

Sub-threshold adapted transmission gate logic has been proposed to increase the variation-resilience without compromising gate performance [7]. A generic block can be used to implement logic functions using TG logic (Fig. 2). Multiplexers and XOR gates, as well as non-inverting gates like OR and AND can easily be implemented. By using nMOS stacking, leakage is reduced and the $I_{\mathrm{on,p}}/I_{\mathrm{off,n}}$ ratio is improved with a negligible performance penalty. Besides variation-resilience in sub-threshold, TG logic also uses less area compared to static CMOS logic in sub-threshold. Since TG combinational circuits use differential input signals, the differential counterpart of each gate is also constructed. Working with differential signals further adds to the robustness of the design. When using TG

Table 1. I_{on} and I_{off} metrics for an nMOS and a pMOS showing the drastic reduction in $I_{\text{on}}/I_{\text{off}}$-ratios at a supply voltage of 0.150 V compared to 1 V

		I_{on}	I_{off}	$I_{\text{on}}/I_{\text{off}}$	$I_{\text{on,n}}/I_{\text{off,p}}$	$I_{\text{on,p}}/I_{\text{off,n}}$
$V_{DD} = 0.150V$	*nMOS*	278 nA	5 nA	54	676	
	pMOS	31 nA	411 pA	75		6.2
$V_{DD} = 1V$	*nMOS*	118 uA	26 nA	4451	59000	
	pMOS	41 uA	2 nA	22478		1576

(a)

	A	B	C
XOR	in1	$\overline{\text{in1}}$	in2
XNOR	$\overline{\text{in1}}$	in1	in2
OR	in1	V_{DD}	in2
NOR	$\overline{\text{in1}}$	V_{SS}	in2
AND	in1	V_{SS}	$\overline{\text{in2}}$
NAND	$\overline{\text{in1}}$	V_{DD}	$\overline{\text{in2}}$
MUX	in1	in2	sel
MUXBAR	$\overline{\text{in1}}$	$\overline{\text{in2}}$	sel

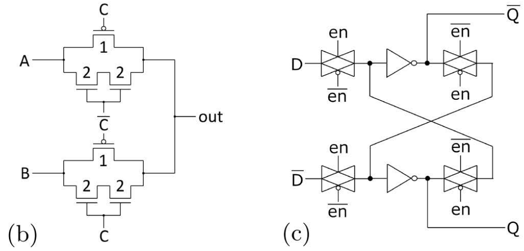
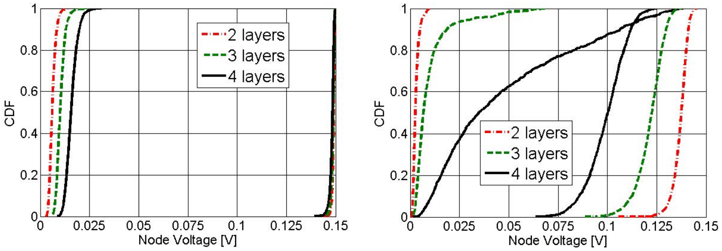


Fig. 2. (a,b) Implementation of logic gates using a TG topology. To reduce leakage, nMOS stacking is used. (c) The basic latch configuration.

logic, the output signal degrades slightly after each gate. To compensate for this loss, the signal is amplified by an inverter after a certain number of gates when implementing combinational circuits. Fig. 3 shows the cumulative distribution function (CDF) of 2000 Monte Carlo (MC) simulations of a logic signal before and after the inverter. The logic signal degrades more with increasing number of logic layers. A high logic level shows a larger spread and degradation after multiple TGs (Fig. 3b) as compared to a low logic level (Fig. 3a), because the pMOS is deeper in sub-threshold and thus more sensitive to variation than an nMOS device for the same supply. An inverter is only able to restore the degraded signal if this signal is within its noise margin. When this is not the case, a logic value can be amplified to the wrong logic value. This is the case when inverters are inserted after 4 layers of logic. But even after 3 layers, the output signal of the inverter shows a larger spread than the input signal, i.e. the spread will increase more after numerous layers of logic and inverters. Only after 2 layers, inverters are able to compensate the signal loss and increased spread due to the use of TGs. Throughout the design inverters are accordingly inserted after a maximum of 2 layers of TG logic.

2.3 Latches and Flip-Flops

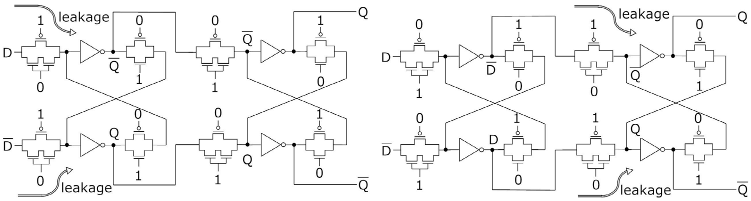
For constructing sequential logic, latches and flip-flops are essential. The basic latch configuration is shown in Fig. 2c. This differential latch uses TGs to cut the feedback loop to enable overwriting the stored value. This uses less energy as opposed to latches that work by overpowering the loop (e.g. a David-Goliath



(a) 0 → 1 conversion by inverter. (b) 1 → 0 conversion by inverter.

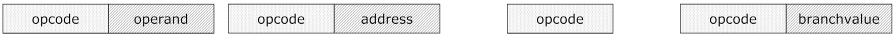
Fig. 3. Low and high voltage levels before and after an inverter for increasing number of logic layers extracted after 2000 Monte Carlo simulations

latch) and also uses less area. Moreover, TG or multiplexer based latches have been shown to be more robust at low supply voltages compared to other topologies [8]. Fig. 4 shows the implementation of the proposed flip-flop, which is a *master-slave* configuration using two latches. The load of its clock signal is an important factor that contributes to the dynamic power consumption. On the other hand, leakage contributes to the static power dissipation when the flip-flops are storing data. Therefore, the TGs in the cross-coupled loops are minimal and non-stacked. This reduces the clock-load by 30% in comparison to when all TGs use nMOS stacking (i.e. using 2 standard latches). Non-stacked TGs leak more than stacked ones. However, the TGs of the cross-coupled loops always have the same logic values on their source and drain sides making leakage no issue. The stability is thus not compromised. The other TGs do have nMOS stacking in order to reduce leakage currents when the flip-flop stores its value.



(a) *clk*-signal is high. (b) *clk*-signal is low.

Fig. 4. A flip-flop with a low activity uses nMOS stacking for TGs prone to leakage. Leakage paths are shown for *clk*-signal high (a) and low (b).



(a) Immediate (b) Absolute (c) Implied (d) Relative

Fig. 5. Representation of the 4 different addressing modes. Each block is 16 bits.

3 Microprocessor Design

The microprocessor designed in this paper is a fully custom designed accumulator-based Harvard architecture, with a physically separated 16-bit data and instruction bus. The implemented instruction set is a subset of the 6502 microprocessor instruction set [9], which includes load, store, register transfer, arithmetic, logical, shift and branch instructions. Four addressing modes are implemented: immediate, absolute, implied and relative addressing (Fig. 5). With immediate addressing, the first 16 bits of opcode are followed by a second 16-bit operand word. In case of absolute addressing, the second 16-bit word holds the address of the data. Implied addressing means that the address is incorporated in the opcode. Relative addressing is only used for branch instructions where the 2$^{\text{nd}}$ word holds the relative shift in the address of the instruction memory.

3.1 Specification

Fig. 6 depicts the main circuit blocks used in the microprocessor. As the architecture is accumulator-based, an operation is always performed on the value in the accumulator register and the result is written back into the accumulator. The processor is in fact a 1-operand machine as only one operand can be specified in the instruction. Apart from the accumulator, the processor is also equipped with X- and Y-registers to store intermediate results. The different processor blocks can be grouped according to their functionality. We distinguish a control unit, processing unit and external memory. The control unit is comprised out of:

- *Opcode register*: To store the first 16 bits of the instruction until the instruction is completed. The opcode specifies the operation of the instruction.
- *Value register*: If an instruction is comprised out of two 16-bit words, the value register will keep the second word until the instruction is completed.
- *Program counter(PC)*: 16-bit to keep the address of the instruction memory.
- *Control*: A finite state machine who's transitions are controlled by the bits of the opcode and the status of the datapath. The control defines when the PC counts, writes the opcode and value registers at the right time and sends correct signals to the datapath depending on the opcode.
- *Branch adder*: Branch instructions stop the sequential execution flow of the program by loading a new value into the PC, calculated by adding the original address to a value given by the value register.

The datapath is comprised out of:

- *Arithmetic logic unit (ALU)*: For the actual data operation the ALU is called by the control. ALU-operations are always executed on the accumulator and the second operand specified in the instruction. The second operand can be the X-register, Y-register, a memory location or the value register.
- *X/Y-register*: A 16-bit register normally used to keep the value of a loop counter. Its value can be incremented or decremented. Store and load operations from the memory are also possible.

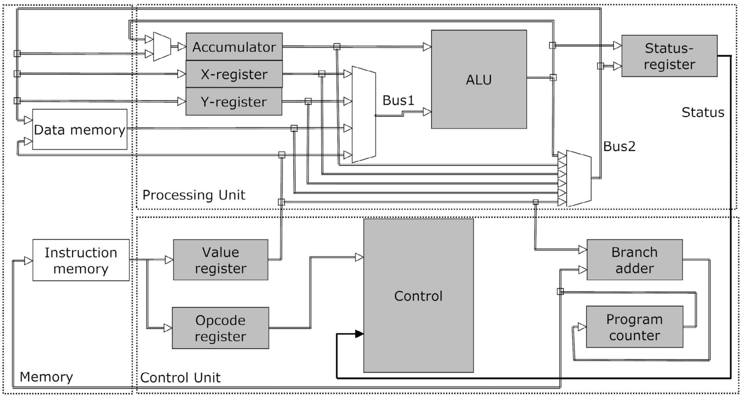


Fig. 6. The microprocessor architecture (processing, control and memory unit)

- *Status register*: After the ALU has completed an operation, a set of status flags are updated depending on the result of the operation. There are four different flags: negative (N), overflow (V), zero (Z) and carry (C). The flags form the status signals by which the datapath informs the control.

The external memory is comprised out of:

- *Instruction memory*: For storing the different instructions of the program.
- *Data memory*: For storing the data needed for and created by the program.

3.2 The ALU

The ALU is responsible for the execution of arithmetic, logical and shift instructions. Therefore it consists of a logical block, a shifter and a logarithmic 16-bit Han-Carlson adder [7] as shown in Fig. 7. In the entire design of the ALU, inverters are repeatedly placed after two layers of logic gates.The implementation of the logical block is obtained by performing a bitwise logical AND, XOR and OR using the generic TG blocks (Fig. 7). The shifter is comprised out of 2 layers of transmission gates, one of which is always on and the other off (right or left shift). There are also 2 multiplexers to insert the appropriate signal on the empty space depending on a shift or rotate operation. The delay of the ALU is given by the critical path delay which is situated in the adder. The worst-case delay is simulated by using the case where a carry-in ripples through to the carry-out, i.e. an increment of Bus1 $= \{1, 1, \ldots, 1\}$. Fig. 8a shows the CDF of this worst-case for 1000 ALUs under intra-die variations. Fitted with a Gaussian distribution, the mean delay of the adder equals 315 ns and the standard deviation equals 21.4 ns. With a 3σ criterion, the ALU's output is reliable after 380 ns. Logical and shift operations (Fig. 8b), take an average of 110 ns to complete with a standard deviation of 13.8 ns. Using the same 3σ criterion, worst-case delay equals 150 ns.

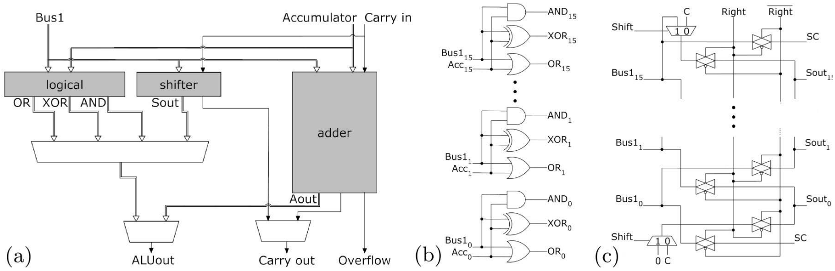
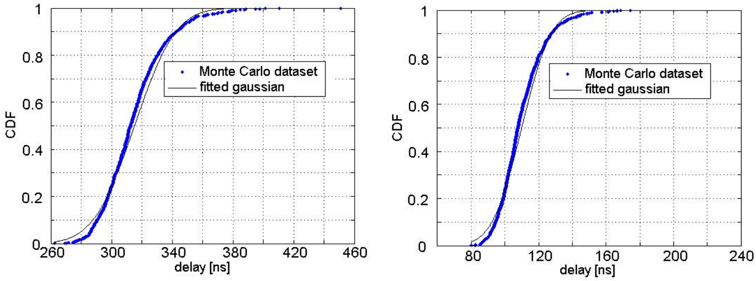


Fig. 7. (a) The internal structure of the ALU with gate level diagram of the (b) logical block and (c) shifter



(a) Adder operations (b) Logical and shift operations

Fig. 8. Delay of ALU operations obtained through 1000 Monte Carlo simulations

3.3 The Registers and the Datapath

16-bit registers are used to store the data, e.g. the X/Y-register and the accumulator. An additional 16-bit register is used to keep the program counter. The register was also subjected to Monte Carlo simulation under intra-die variations. The propagation delay T_{c-q} has an average value of 9.47 ns with a σ of 3.07 ns. The worst-case propagation delay is then set at 18.68 ns.

3.4 Control

The control determines how an instruction is executed by going through the processor cycle (Fig. 9a). There are basically three steps: a fetch step where the instruction (opcode and optional value) is fetched from the memory, an execute step where the ALU executes an operation and a write back step where the result is written back into the accumulator. The control is then a finite state machine (FSM) which is build as a sequential circuit comprised out of combinational logic and flip-flops. The FSM is implemented as a Moore-type FSM, where each state corresponds to an equal delay of one clock period (250 ns). Fetching the instruction and writing back the result can thus be performed in a single state since the worst-case delay is lower than 1 clock period. The execute

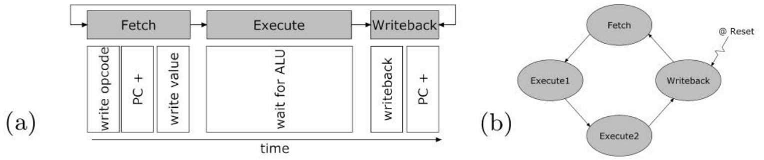


Fig. 9. (a) The processor cycle corresponds to (b) four states with equal delay. With this FSM, the throughput is 1 MIPS at a clock frequency of 4 MHz.

step on the other hand needs to be represented by two states as the worst-case delay of the ALU is larger than one clock period. It takes two clock periods (500 ns) for the worst-case delay of the ALU (380 ns) to be lower than the execute step. The execute step is accordingly split into two states with each a delay of 250 ns. As a result, there are 4 states in total to represent the processor cycle (Fig. 9b). At a clock frequency of 4 MHz, 1000 ns is required to complete one instruction, this corresponds to 1 MIPS (Mega Instruction Per Second) . Working with this approach allows for a robust sequential circuit which is needed in the sub-threshold design. There are two D-flip-flops required in the FSM design to incorporate four states. This small number of states (and flip-flops) results in only one logic layer between consecutive flip-flops (state transition logic).

3.5 2-Stage Pipelining

A common design technique to boost the performance of a processor is pipelining. By introducing pipeline stages, throughput is increased, which in turn will affect the power consumption as well. Not only the dynamic power will increase because of the higher activity, static power consumption also increases due to the additional hardware needed for the pipelined implementation (extra flip-flips and registers). The trade-off between speed, dynamic energy and leakage power favours pipelining in super-threshold standard CMOS. To prove that pipelining is also advantageous in TG logic sub-threshold design, a second microprocessor is designed using a 2-stage pipeline. Fig. 10a shows the timing diagram of the standard microprocessor with a throughput of 1 MIPS, while Fig. 10b shows the diagram for the pipelined version. When the ALU is performing an operation, the previous result can be written back and the next instruction can already be fetched out of the instruction memory.

4 Simulation Results

In this section the two different microprocessors are compared on power consumption, energy per instruction and Energy-Delay-Product (EDP). The throughput of the standard microprocessor μP_1 equals 1 MIPS, while the throughput of the pipelined microprocessor μP_2 equals 2 MIPS (not considering branch stalls).

The microprocessor was designed in a 90 nm CMOS technology using TG logic. The clock is set at 4 MHz and V_{DD} at 150 mV. The implementation of the ALU,

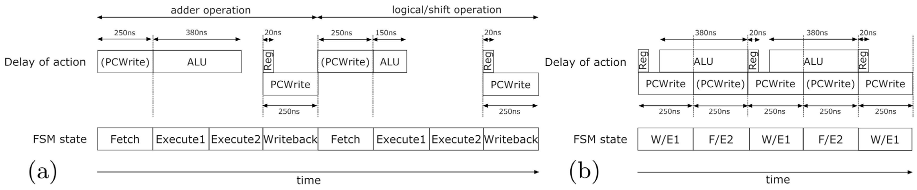


Fig. 10. Timing diagrams for (a) the standard and (b) the pipelined microprocessor. PCWrite involves writing the correct instruction address into the PC.

PC, branch adder, buses and status register are the same for both processors and all subjected to MC simulation. Results are obtained by completely simulating the two microprocessor topologies with a spice-level simulator, where the instruction and data memory are simulated in a behavioral way using verilog-a. Due to spice-simulation with high transistor count (4000) only small test programs were applied. To obtain a fair comparison, both a program with and without branch instructions is simulated on both microprocessors. The first program calculates the average of 4 numbers, which is a small program containing 10 instructions. The second program selects the largest number out of 4 given numbers. This program is larger and has branch instructions.

A power consumption of 0.74 μW is obtained together with an energy per instruction of 0.74 pJ for μP_1 (Fig. 11a). The Energy-Delay-Product (EDP), the Figure Of Merit that balances the importance of both energy and performance, equals 0.74 pJ$\times\mu$s. A more energy efficient design is obtained by using pipelining. The energy per instruction of μP_2 is reduced to 0.48 pJ. This, combined with the increased throughput, allows μP_2 to have an EDP of 0.23 pJ$\times\mu$s. Fig. 11b shows the power distribution of both microprocessors. Fig. 11c gives a com-

	total power [μW]	energy/instruction [pJ]	EDP [pJ$\times\mu$s]	throughput [MIPS]	energy efficiency [GIPS/W]
μP_1	0.74	0.74	0.74	1.0	1400
(a) μP_2	0.91	0.48	0.23	2.0	2200

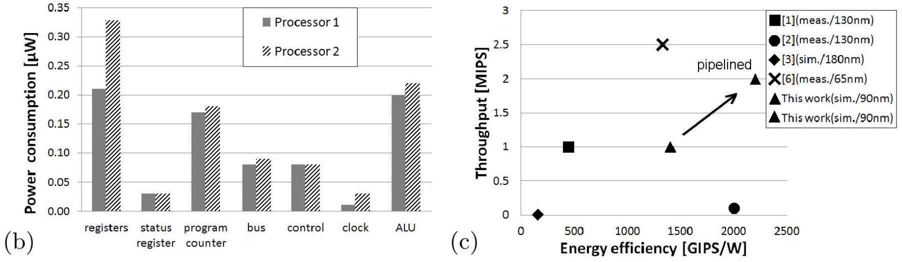


Fig. 11. (a,b) Specifications and power distribution under a V_{DD} of 150 mV and a 4 MHz operating frequency. (c) State-of-the-art comparison of throughput and energy efficiency of ultra-low-power designs (only processor core energy).

parison of throughput and energy efficiency of the processors compared to other state-of-the-art works. Only the processor core energy is taken into account. Also reported is the used technology node and whether results were obtained through simulation or measurement. The proposed processor achieves better combined energy efficiency and throughput. [2] has a similar energy efficiency but has considerably less throughput as compared to this work. [6] has a higher throughput, but has an energy efficiency about 2× less than this work. A variation-resilient sub-threshold design using TG logic instead of static CMOS logic facilitates the design of high performance and energy efficient processors.

5 Conclusion

This paper presented an ultra-low-power 16-bit microprocessor simulated in a 90 nm CMOS technology, operating in the sub-threshold region by lowering V_{DD} down to 150 mV. The microprocessor achieves a computational performance of 2 MIPS at an energy per instruction of 0.48 pJ. The first technique used to design this variation-resilient and highly energy-efficient microprocessor is differential transmission gate logic, because it appears to be excellent for robust sub-threshold logic design. Pipelining is introduced in order to increase the throughput. Moreover, it is shown that employing pipelining on circuits operating in the sub-threshold region has a very beneficial impact on the energy consumption per instruction, resulting in a better than state-of-the-art energy efficiency.

References

1. Zhai, B., et al.: A 2.60pJ/Inst Subthreshold Sensor Processor for Optimal Energy Efficiency. In: IEEE Symposium on VLSI Circuits, pp. 154–155 (2006)
2. Hanson, S., et al.: Exploring Variability and Performance in a Sub-200-mV Processor. IEEE Journal of Solid-State Circuits, 881–891 (April 2008)
3. Jin, W., et al.: A 230mV 8-bit Sub-threshold Microprocessor for Wireless Sensor Network. In: IEEE Int. Conf. on VLSI and System-on-Chip, pp. 126–129 (October 2011)
4. Wang, A., et al.: Optimal Supply and Threshold Scaling for Subthreshold CMOS Circuits. In: IEEE Computer Society Annual Symposium on VLSI, pp. 5–9 (April 2002)
5. Sze, V., Chandrakasan, A.P.: A 0.4-V UWB Baseband Processor. In: ACM/IEEE Int. Symp. on Low Power Electronics and Design, pp. 262–267 (August 2007)
6. Pu, Y., et al.: An Ultra-Low-Energy Multi-Standard JPEG Co-Processor in 65nm CMOS With Sub/Near Threshold Supply Voltage. IEEE Journal of Solid-State Circuits, 668–680 (March 2010)
7. Reynders, N., Dehaene, W.: A 190mV supply, 10MHz, 90nm CMOS, Pipelined Sub-Threshold Adder using Variation-Resilient Circuit Techniques. In: IEEE Asian Solid State Circuits Conference, pp. 113–116 (November 2011)
8. Jin, W., et al.: Robust Design of Sub-threshold Flip-Flop Cells for Wireless Sensor Network. In: IEEE Int. Conf. on VLSI and System-on-Chip, pp. 440–443 (October 2011)
9. Commodore Semiconductor Group (1985), http://www.6502.org/

Run-Time Measurement of Harvested Energy for Autarkic Sensor Operation

Dimitris Bekiaris, Ioannis Kosmadakis[2], George Stassinopoulos[1], Dimitrios Soudris[1], Theodoros Laopoulos[2], Gregory Doumenis[3], and Stylianos Siskos[2]

[1] National Technical University of Athens (NTUA), Greece
[2] Department of Physics, Aristotle University of Thessaloniki (AUTH), Greece
[3] GDT S.A.
{mpekiaris,dsoudris}@microlab.ntua.gr,
{siskos,laopoulos,ikosm}@physics.auth.gr,
stassin@cs.ntua.gr, greg@gdt.gr

Abstract. Harvesting energy from the environment in order to provide sensors with autarky entails many challenges. Foremost is the ability to match the energy uptake and storage, so that the maximal yield possible is attained. This involves sophisticated analogue circuitry, to continuously match the harvesting element. It also involves digital logic exploiting features of the analogue modules, to quantify and control the energy budget at run-time. The overall goal was to minimize periods of energy starvation, as well as periods where harvesting is not possible, due to limited storage capacity. The present paper introduces an on-chip vibration harvesting unit fabricated with the 0.35um AMS CMOS technology, feeding sensors in the presence of extremely volatile energy environments. It demonstrates both analogue and digital parts integrated on a *3.06mm²* chip, as well as their coordination, targeting autarkic sensor operation.

Keywords: Energy, Harvesting, Vibration, SEPIC, Ripple counter, SPI.

1 Introduction and Related Work

Exploiting energy harvested in erratic and unpredicted environments has been addressed in a variety of ways and applications, highlighting different flavours of energy sources [1-6]. In [3], the tight coupling between temporary power levels and achievable voltages can allow operation of a DSP within a wide spectrum of harvesting intensity. This tight coupling amounts to changing the supply voltage, while controlling a commensurate DSP frequency at the same time.[1]

[1] Research activities that led to the results reported here were co-financed by Hellenic Funds and the European Regional Development Fund (ERDF) under the Hellenic National Strategic Reference Framework (NRSF) 2007-2013, contract MICRO2-42 “Development of Innovative Sensor Systems offering Distributed Intelligence – MEMSENSE”, within the Program “Hellenic Technology Clusters in Microelectronics”.

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 185–193, 2013.

In this work, we envisage a looser coupling, made possible by state-of-the-art power conditioning circuitry, use of temporary power storage and intelligence in the power adaptation loop, provided by a fully fledged digital part. Information on the energy actually harvested is supplied digitally in a microcontroller-hosted algorithm [7]. The latter has a dual task, i.e. to prevent starvation of the energy stored to enable continuous operation, while ensuring at the same time that adequate capacity is provided, so that possible peaks of harvesting potential are absorbed.

We address erratic, vibration-based energy harvesting, in uncontrolled environments and semi-periodic load servicing appearing in autonomous sensors tasks. Battery storage is available but to a limited degree and has to be optimally exploited, depending on the targeted application. This is performed through a tightly coupled analogue-digital design, able to absorb and quantify the harvested energy at run-time.

The run-time quantification of the energy actually harvested comprises one of the main contributions of this work and it is performed through a 16-bit Ripple counter, bridging the analogue with the digital part. The count of absorbed energy, along with crude information about the state of the storage battery, can be led to a load controlling algorithm, which constitutes the procedural part of our self-contained module and it is presented in detail in [7].

In this paper, key design choices and implementation details are outlined in Sections 3 and 4 respectively, whereas on-chip measurements and post-layout simulation results are presented in Section 5. Concluding remarks summarizing the contributions of this work are outlined in Section 6.

2 Overall System Description

The diagram of Fig. 1 presents the architecture of the proposed Autarky module, as a self-contained component. It can be fed by multiple external energy harvesting devices and may serve a variety of possible load configurations, via a load battery (LB in Fig. 1), used as intermediate energy buffer. Under mild operating conditions, as exposed below, "intelligence" in absorbing and controlling incoming energy ensures autarky under graceful degradation or upgrade of service, under the presence of wild environmental fluctuations of the harvesting potential. In this work, we focus on the on-chip design and implementation of the proposed Autarky module, harvesting energy coming from vibrations. However, the specific system can be used in applications exploiting either solar or alternative environmental energy sources.

The system's analogue circuitry, which is responsible for capturing energy and controlling its flow, rectifies power harvested from an off-chip vibration source and it converts the resulting DC level, thus feeding storage and load. A capacitor of some μFs, namely C in Fig. 1, acts as temporary storage, to smooth out the erratic source behaviour. The control part of the circuitry has the task to gracefully adjust consumption and thus the service provided, in the presence of short and medium term fluctuations of the power uptake from the environment. It also supports this power uptake by ensuring available storage capacity, through enhancement of the service provided.

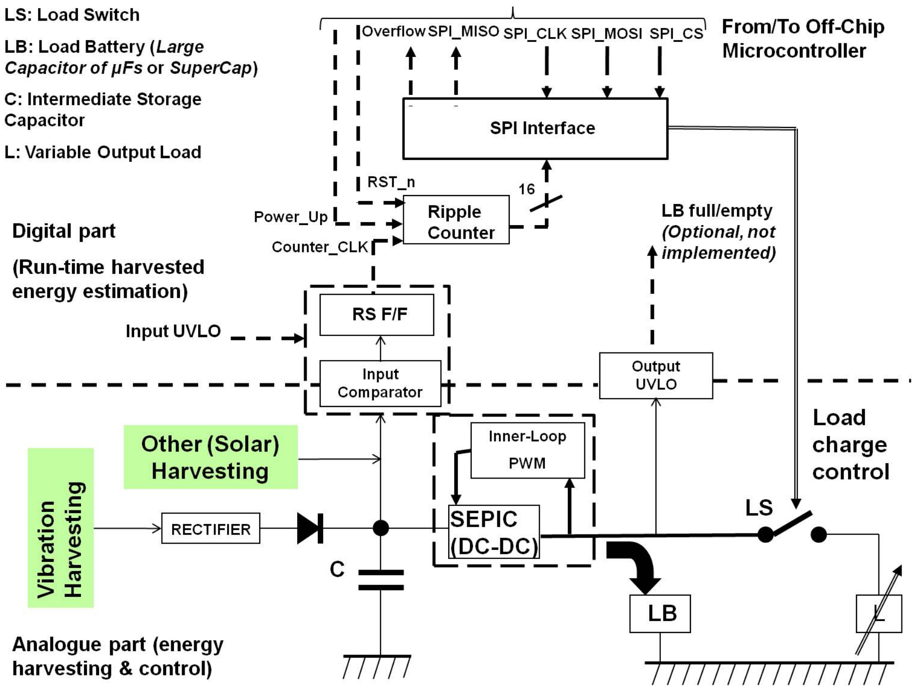


Fig. 1. Block diagram of the proposed on-chip Autarky Module

This paper puts emphasis on the progress achieved in monitoring and quantifying harvested energy at run-time. Through an on-chip mixed-signal (analogue - digital) design, as shown in Fig. 1, we ensure that a counter effectively measures the energy absorbed by the power module. This information, capable of responding to short-term fluctuations of harvesting potential, greatly enhances the adaptability and reaction speed of the developed on-chip harvesting unit.

3 Matching, Conditioning and Absorbing Harvested Energy

The analogue part, which is responsible for input rectification and load battery storage, is designed to match the impedance between transducer and rectifier, as well as between rectifier and DC-DC converter. Impedance matching aims at eliminating energy loss and signal propagation delay between these stages. The input of the left of Fig. 1, terminating the energy harvesting device, has to match the U-I characteristics and the spectral behaviour of this device as power source.

There is a two sided view on the overall configuration of the analogue part, as shown in Fig.1. One concerns matching to the vibration source, the second offering a match to the load. The Rectifier is driven by an AC input provided by the micro-generator and its output feeds the DC-DC converter. A full-wave rectifier offers a relatively higher efficiency than half wave rectifiers and voltage doublers. However, a

MOSFET bridge rectifier in series with an “active” diode was selected for synchronous rectification. This ensures low-power operation, combined with the maximum achievable efficiency. The designed rectifier is able to operate with AC input signals of 2-5V in amplitude and 50-150Hz in frequency. The rectifier is bypassed in case of using a more traditional harvester, like a solar cell, as shown in Fig. 1, fed with DC input.

During DC-DC conversion, the input voltage level is converted to the desired one, by temporarily storing the collected input energy and subsequently releasing it to the output. This offers higher power-efficiency, compared to any linear type voltage converter. Hence, a SEPIC (Single-Ended Primary-Inductor Converter), whose schematic is given in Fig. 2, is selected, due to its up/down level conversion capability.

The SEPIC converter uses two inductors to store incoming energy and a capacitor to isolate the input from the output. The SEPIC designed in this work is based on a fixed frequency, voltage-mode Pulse-Width Modulation (PWM) controller. In order to reduce the required inductance of the SEPIC, the operating frequency of the PWM circuit is in the order of hundreds of MHz. The regulated output voltage across the output capacitor (C_{out} of Fig. 2) of the designed converter, charging the battery of Fig. 1 (LB), is set to 4.2V.

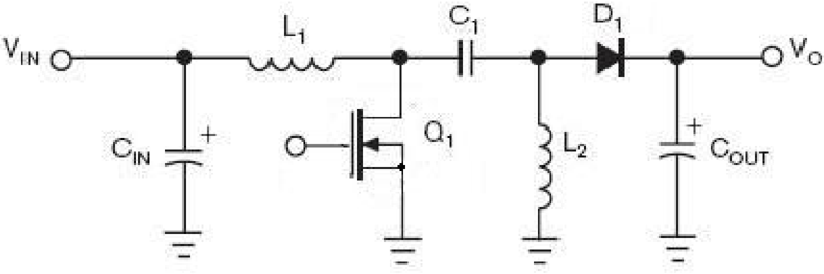


Fig. 2. Schematic diagram of the SEPIC DC-DC converter

The SEPIC’s output is dedicated to load the battery of Fig. 1 (LB), to store the energy harvested and absorbed into C. This function is in the context of storing the captured energy from vibrations. Therefore, the SEPIC is enabled or disabled, depending on the voltage level across C. This is monitored by an Under-Voltage Lockout (UVLO) circuit, involving voltage comparators and an RS flip-flop. The UVLO disables SEPIC, if the voltage across C is below the threshold V_{ref_stop} (*1.152V*).

Then, the system enters a “power-saving” mode. The converter is enabled again, when the capacitor’s C input voltage rises beyond the desired level, namely V_{ref_start} (*2.296V*). The UVLO’s comparator generates a negative edge pulse, which resets as RS flip-flop, inverting the pulse and triggering a digital ripple counter. This pulse triggers a ripple counter, the output of which is an indication for the energy actually harvested and absorbed in the LB, throughout SEPIC. Optionally, the same UVLO, tuned to higher $V_{ref\_start}/V_{ref\_stop}$, could be used to monitor the battery’s voltage level.

4 Harvested Energy Estimator

The main contribution of this paper, which differentiates it, to the best of our knowledge, from most of the relative published works, is focused on the run-time quantification of harvested energy events, generated from vibrations. These are captured at run-time by the analogue circuitry of Section III, which monitors the charging and discharging of the intermediate capacitor C, feeding the SEPIC. However, a real-time on-chip estimation of the harvested energy also requires a digital datapath module, endowed with the possibility of efficiently mapping each charging of C to a specific action.

Thus, our intention was the design of a low-complexity digital circuit, able not only to update, but also to accumulate the times C has been charged to V_{ref_start}. For such purposes, a 16-bit counter was selected, due to its tolerable silicon area and its ability to map incoming energy events to counts. The selected bit-length combined accuracy with low complexity.

Each time the voltage across C reaches V_{ref_start}, a pulse is generated from the input UVLO's comparator, setting the RS flip-flop, which produces the counter's clock and triggers the counter. When C is discharged to V_{ref_stop}, the RS flip-flop is reset and this is the end of the clock's pulse width.

Hence*, the arrival of a new rising clock edge is determined by the capacitor's voltage level and thus, by the amount of energy actually harvested.* The total number of charge cycles of capacitor C is maintained and updated by the counter.

As the proposed system is likely to operate in energy-starving environments, long-term idle periods between consecutive counts are possible. Therefore, the counter's clock frequency is expected to be be relatively low (around KHz) and of unpredictable duty cycle. Such a constraint, imposed by the environment, led to the selection of a Ripple scheme, instead of well-known low-power counter designs (e.g. Gray). However, the proposed counter comes along with a clock-gating circuitry, also depicted in Fig. 3. The Ripple's clock should be gated by the microcontroller during idle periods between consecutive vibration harvesting events, to prevent a counter's increment while attempting to read its current value.

Such idle periods leave sufficient timing slack to the microcontroller for reading the counter's value. Therefore, a bit-serial transfer of the counter's bits was selected, instead of a bit-parallel interface incurring extra silicon area and power, as well as a larger package, due to the additional I/O pads.

Hence, a Serial Peripheral Interface (SPI) module is selected to transmit the accumulated counts to the off-chip microcontroller. The specific component is differentiated from a typical SPI slave, as SPI_MOSI input is a control signal, instead of carrying data sent by the master (microcontroller).

In Fig. 3, the SPI_CLK input pin, controlled by the microcontroller, clocks the internal shift register, whose input data pins are controlled by a 2-to-1 multiplexer. The latter is controlled by SPI_MOSI, selecting either the 8 LSB ('0') or the 8 MSB ('1') counter bits. Once SPI_CS is set, the 2-1 MUX output is loaded into the shift register at the positive edge of the SPI_CLK clock. When SPI_CS is deactivated, the shift register starts to serially transfer the previously loaded 8 LSB or MSB counter bits through SPI_MISO, according to Fig. 3. In Fig. 4, the waveform demonstrates the counter's 8 LSB transfer, requiring 9 SPI_CLK cycles. Consequently, the counter's reading is completed after 18 such clock cycles.

Regarding Overflow, namely the digital module's second output, it is set when the counter's bit 15 (counter's *Q15* output, according to Fig. 3) rises, signifying *32768* captured incoming energy events. Optionally, the microcontroller could be exploited, in order to reset the counter and flush the already captured charges of C to $V_{\mathrm{ref\_start}}$, by connecting Overflow to one of its interrupt input pins.

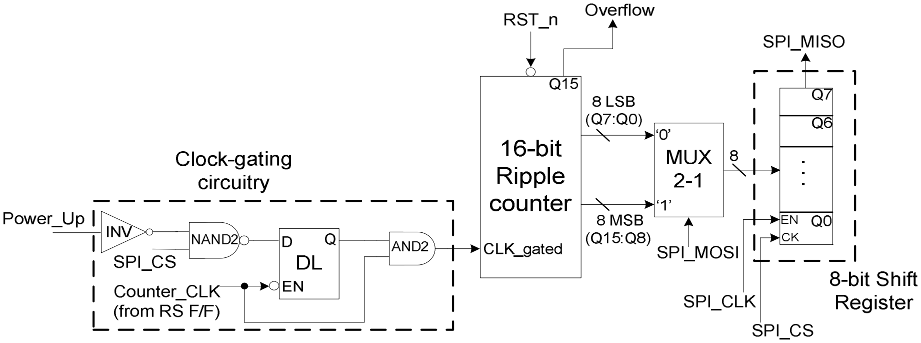


Fig. 3. The system's digital part demonstrating the 16-bit clock-gating Ripple counter, coupled with the proposed customized SPI module

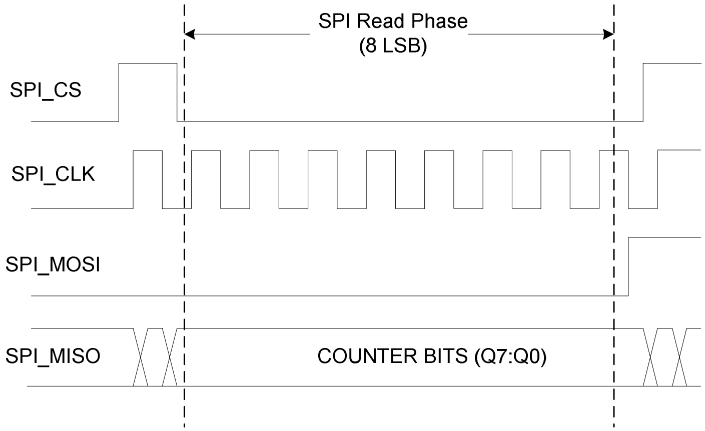


Fig. 4. The SPI read phase of the Ripple counter's 8 LSB bits

5 Measurements and Experimental Results

Post-layout simulations were performed in Cadence Virtuoso ADE [10], using Spectre simulator and operating at typical conditions. The analogue core's (ASC_A) voltage was 3.3V, while the corresponding I/O pads operated at 4.2V. Both the digital core's (ASC_D) standard-cells and its pads work at 1.8V. For the fabrication, a 44-pin JLCC package was selected to fit to the *3.06mm2* chip's silicon area.

The post-layout simulations are coupled with measurements validating the chip's functionality, while harvesting energy from vibrations. These are produced by a MIDE V25W [9] piezoelectric harvester at 90Hz, feeding the rectifier's input.

Based on this setup, we demonstrate the rectifier's output in Fig. 5. The input generated by vibrations is the blue-colored waveform, whereas the narrow-width red-colored pulse is the input UVLO's output, driving the Ripple counter's clock, as Fig. 1 shows. Also, in Fig. 6, the SEPIC's output is given to validate that harvested energy is transferred from the rectifier to the DC-DC converter and then to the load battery, LB.

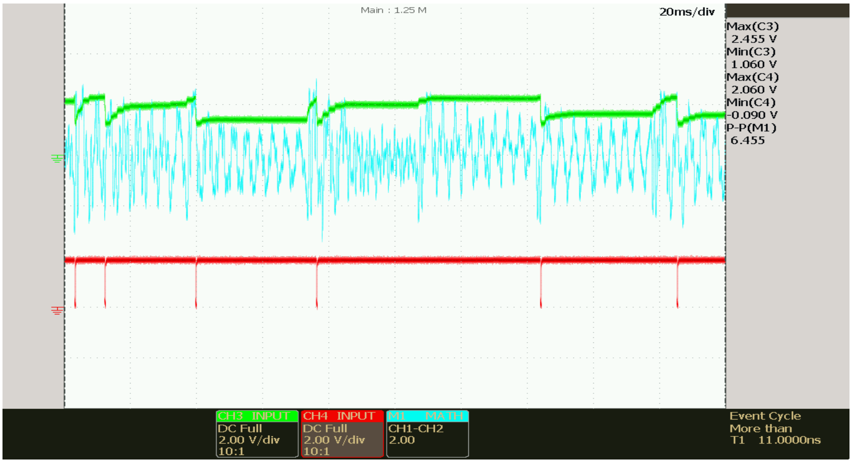


Fig. 5. The on-chip measured Rectifier and input UVLO outputs

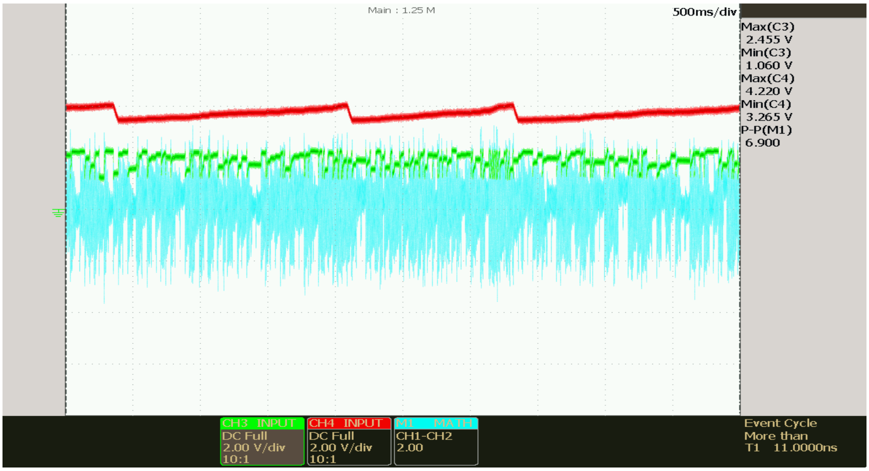


Fig. 6. The SEPIC output, depicted by the red-colored waveform

These measurements are in agreement with the conducted post-layout simulations results, based on the two scenarios involving SEPIC (enabled/disabled by the input UVLO). The results demonstrate the system's total power dissipation, when the rectifier is fed with an AC input of 3.5V and 100Hz.

The conducted total power *when SEPIC is enabled* reaches *100μW. When the input UVLO disables SEPIC*, the average consumption falls down to *2μW*, including the digital part's share. The latter is negligible, due to the counter's low operating voltage (1.8V) and clock frequency, around 1KHz. The system's maximum (%) efficiency ($V_{SEPIC,out}/V_{Rectifier,in}$) derived from these simulations reached 73%.

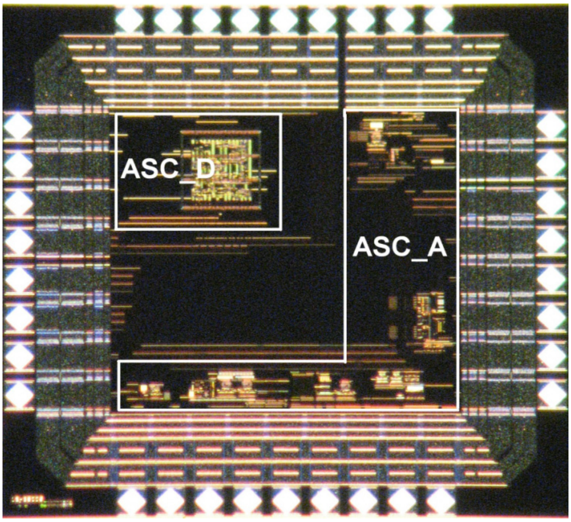


Fig. 7. Microphotograph of the *3.06mm²* chip, fabricated with the 1.8V AMS 0.35um CMOS technology

6 Conclusion, Discussion and Hints for Future Work

The present work introduces an on-chip Autarky Module fabricated with the 0.35um AMS technology, dedicated to power sensors and relative autonomous devices. Its key feature is the run-time quantification of harvested energy, while operating in erratic environments. This is achieved through a digital counter exploiting state-of-the-art analogue circuitry, to estimate the energy absorbed in real time.

References

1. Rajasekaran, A., et al.: Buck-Boost Converter for Sensorless Power Optimization of Piezoelectric Energy Harvester. IEEE Transactions on Power Electronics 22(5), 2018–2025 (2007)
2. Priya, S., Inman, D.J. (eds.): Harvesting Microelectronic Circuits. Energy Harvesting Technologies. Springer (2008)

3. Amirtharajah, R., et al.: Self_powered Signal Processing Using Vibration-Based Power Generation. IEEE JSSCC (May 1998)
4. Chee, Y., et al.: PicoCube: a 1 cm^3 sensor node powered by harvested energy. In: Design Automation Conference (DAC), Anaheim, USA (2008)
5. Carli, D., et al.: An effective multi-source energy harvester for low power applications. In: DATE 2011, Grenoble, France (2011)
6. Aktakka, E., et al.: A Self-Supplied Inertial Energy Harvester with Power Management IC. In: IEEE ISSCC 2011, San Fransisco, USA (2011)
7. Vergados, D., Stassinopoulos, G.: Adaptive Duty Cycle Control for Optimal Stochastic Energy Harvesting. Journal of Wireless Personal Communications (November 2011)
8. Sharma, V., et al.: Optimal energy management policies for energy harvesting sensor nodes. IEEE Transactions on Wireless Communications 9(4), 1326–1336 (2010)
9. MIDE V25W harvester, `http://www.mide.com/products/volture/v25w.php`
10. Cadence Virtuoso ADE User Guide, `http://www.cadence.com`

Low-Power Delay Sensors on FPGAs

Panagiotis Sakellariou and Vassilis Paliouras

Electrical and Computer Engineering Department
University of Patras, Greece

Abstract. This paper investigates design and implementation issues of all-digital delay sensors implemented on FPGAs. The delay sensors discussed here are suitable for monitoring of digital systems during operation. Two topology families are studied. Focusing on power reduction in addition to quality of measurement, corresponding power dissipation reduction techniques are introduced, fully characterized by measurements on actual FPGA hardware implementations. The proposed delay sensors derived by means of an introduced design method, reduce power consumption by 31% for cases of practical interest.

1 Introduction

A delay sensor is a circuit able to detect time deviation in an electronic system. The delay fluctuations have an impact on both functionality and power consumption of a digital system. Many factors lead to delay variation and, through that, to functional failure. Some of the most important factors are the thermal conditions of the circuit, IR drop and voltage variation, switching activity, heavy computing load in case of processors, manufacturing defects, circuit density, circuit neighborhood, parasitic capacity and so on [1][2]. In most circuits it is possible to detect the critical path of a circuit in the pre-synthesis phase [3]. In modern technologies the maximum-delay critical path may change during system operation rendering delay variation detection useful. Apart from circuit functionality, delay variation leads to power consumption increase [4].

In the approach adopted in this paper, delay circuits monitored based on the principle of proximity, i.e., by assuming that similar conditions affect neighbouring circuits in a similar matter. Further use can be found by observing that there is linearity between delay measurement and thermal conditions. In fact using a converting function, it is possible to correspond the delay measurement to temperature indication [4][5][6].

In this paper two topologies of delay sensors are studied and characterized. Design improvements for low power are introduced along with a design space exploration which quantifies power consumption results and the resolution of measurement.

The study presented here is focused on FPGAs. Since FPGAs can speed up the prototyping process, they are suitable for verification and acceleration. In addition, state-of-the-art indicates the use of FPGAs as reconfigurable hardware units able to accelerate different processes. The proposed all-digital sensors can

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 194–204, 2013.

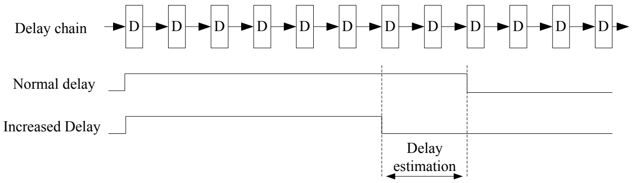


Fig. 1. Delay detection on Feed-forward topology

be implements either using FPGAs LUT or as a hard block on ASIC technology. Moreover, the proposed all-digital sensors can assure the functionality on FPGAs while offering an indication of power consumption variance due to delay variance.

The remainder of this paper is organized as follows. In Section 2 the delay sensor topologies and the principles are presented. Section 3 presents the designs and the power consumption of delay sensors. Section 4 introduces the optimal low power consumption designs. In Section 5, the results of both resolution and power consumption are highlighted. The paper concludes in Section 6.

2 Delay Sensor Topologies Principles

Digital delay sensors can be categorized according to their operating principle. A delay sensor comprises two components, one that produces a signal sensitive to delay variation, and a second that quantifies the delay by observing the delay-sensitive signal. The operating principle of the former component is that a signal passes through several identical elements that introduce a propagation delay to it.

We study two topologies for the structure of the delay-sensitive part, namely the feed-forward topology and the feedback loop topology. In the feed-forward topology, the delay sensitive signal is produced by a non-feedback circuit. The principle of the feed-forward is topology presented in Fig. 1. Once the system is enabled, a single pass suffices to quantify the delay. In the feedback-loop topology, a signal traverses a feedback loop for a certain period of time. The advantage of the second topology is that it reuses circuit elements to form a loop in order to magnify the delay effects and, therefore, it occupies significantly smaller area on the die. The second topology can be considered as a folded implementation of the first [7].

Delay sensors that adopt the feed forward topology can exhibit greater variance and lack of the homogeneity, resulting in substantially increasing the cost of hardware. Furthermore, the physical placement of a sensor adopting the particular topology can be considered challenging in order to achieve proximity to specific units of interest, in the monitored system. This architecture can be considered as a very long chain of identical delay elements. In order to be physically close to the functional units, great effort is required to achieve proper placement. Therefore, an approach is to implement multiple crossings of the delay chain through the functional unit of interest can be adopted.

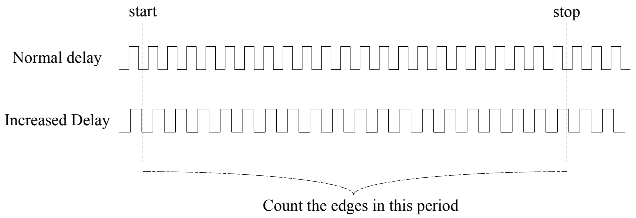


Fig. 2. Delay detection on Feedback-loop topology

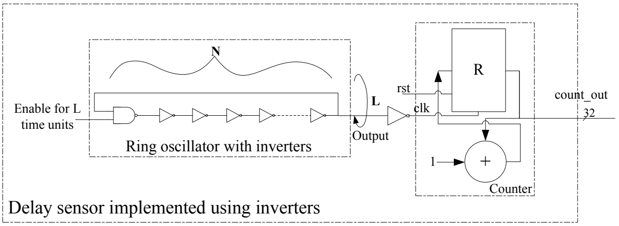

N
L
Enable for L time units
Ring oscillator with inverters
Output
rst
clk
R
count_out
32
1
+
Counter
Delay sensor implemented using inverters

Fig. 3. Feedback loop Design

Feedback loop sensors require a small number of identical delay elements connected in a loop. The loop topology magnifies the delay sufficiently enough, so that a counter or another counting structure is able to measure this delay. The operating principle of the feedback loop topology is clarified in Fig. 2. Due to the small number of delay elements, the design is compact and the placement in the monitored system is simplified. Furthermore, this architecture allows the implementation of large numbers of sensors in a system, i.e., it allows the implementation of a multi-sensor system providing higher accuracy and even the ability of delay variation spread prediction in the system, in some cases.

3 Delay Sensor Designs

3.1 Feedback-Loop Delay Sensor Design

Ring oscillators can be categorized as feedback-loop delay sensors, and can be used as clock generators, delay sensors and thermal sensors [6][8][9]. If a delay is added to the ring then the generated frequency decreases, hence, in a particular time frame, the number of the clock edges would be less. Using a fixed-length counter, it is possible to measure the frequency of the clock generated by a ring oscillator. The counter is fed with the generated clock which works as a gated clock. Thus, every edge is detected while there is no need for a high-speed sampling mechanism. The only constraint is that the generated clock must be slower than the critical path of the counter determined by the length of the counter in bits. Using the datasheet [10] of the target FPGA device the maximum frequency of the counter is quantified. The organization of the feedback-loop approach is depicted in Fig. 3, comprising the ring oscillator and a counter. Each element is implemented using LUT while the counter uses a DPS48 on Virtex-5 technology.

The operation frequency of the counter is determined by the ring oscillator through the number of taps N. The operation time interval denoted as L, contains the number of complete oscillations of the ring oscillator signal and it is defined by an FSM clocked by the system clock. Eqs. (1)–(3) present the dynamic energy consumption of the delay sensor.

$$\begin{aligned} P_{\text{ring}} &= N \cdot P_{\text{inv}} = N \cdot \frac{C_{\text{inv}} \cdot V_{DD}^2}{2 \cdot N \cdot t_p} = \frac{C_{inv} \cdot V_{DD}^2}{2 \cdot t_p} \Rightarrow \\ E_{\text{ring}} &= L \cdot t_{\text{clk}} \cdot \frac{C_{\text{inv}} \cdot V_{DD}^2}{2 \cdot t_p} \end{aligned} \tag{1}$$

$$P_{\text{counter}} = \frac{C_{\text{counter}} \cdot V_{DD}^2}{2 \cdot N \cdot t_p} \Rightarrow E_{\text{counter}} = L \cdot t_{\text{clk}} \cdot \frac{C_{\text{counter}} \cdot V_{DD}^2}{2 \cdot N \cdot t_p} \tag{2}$$

$$\begin{aligned} E_{\text{delay sensor}} &= E_{\text{ring}} + E_{\text{counter}} + E_{\text{fsm}} \Rightarrow \\ E_{\text{delay sensor}} &= L \cdot t_{\text{clk}} \cdot \frac{C_{\text{inv}} \cdot V_{DD}^2}{2 \cdot t_p} + L \cdot t_{\text{clk}} \cdot \frac{C_{\text{counter}} \cdot V_{DD}^2}{2 \cdot N \cdot t_p} + E_{\text{fsm}} \end{aligned} \tag{3}$$

The total required energy is composed of the energy consumed by the ring oscillator E_{ring}, the energy consumed by the counter and the control denoted as E_{counter} and E_{fsm} respectively. The energy consumption of the ring oscillator is proportional to the time interval length L, to the system clock t_{clk} and to the square of the source voltage denoted as $V_{DD}^2.C_{inv}$ and t_p represents the capacity and the propagation delay of an inverter respectively. The energy consumption of the counter is inversely proportional to the number of N stages of the ring oscillator because of the generated frequency and proportional to the L factor. The effective capacity is denoted as C_{counter}.

The two factors N and L determine the power consumption and at the same time, the resolution of the delay sensor. The resolution is proportional to the maximum counted edges given by

$$Counter_{\text{edges}} = \frac{L \cdot t_{\text{clk}}}{2N \cdot t_p} \tag{4}$$

Hence, there is a relation between sensor resolution and its power consumption. A design space exploration takes place in order to find a (N, L) point where the desired measurement resolution is achieved with the lower power consumption.

3.2 Feed Forward Delay Sensor Design

In the feed-forward topology, a cascade of delay elements increases the propagation delay of a signal. The design presented resembles the principle of [11]

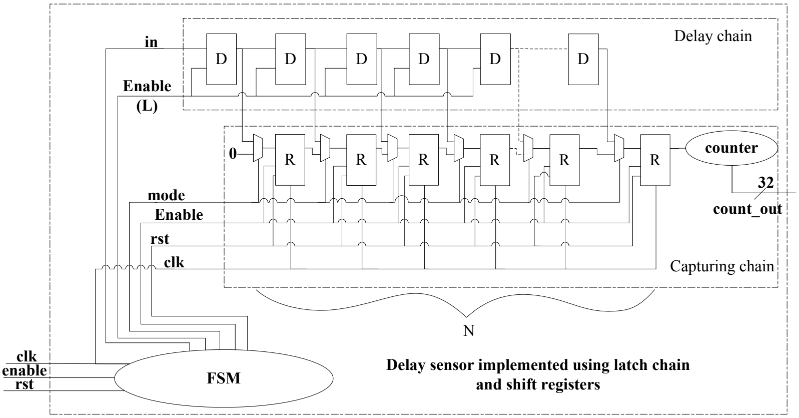


Fig. 4. Feed Forward Design

which refers to a transistor-level implementation. By sampling the delay elements output every few stages, the signal delay propagation is quantified. If the delay increases, then the signal propagates through fewer elements in a specific time interval. Measuring the number of elements through which the signal propagates gives an estimation of delay. As delay elements in this structure, it is possible to use buffers or transparent latches or any structure which does not alter the signal value. By capturing the signal propagation path length, storing into registers and reading the outputs of the registers gives a delay estimation. The organization of the feed-forward approach is depicted in Fig. 4.

The number of minimum N taps is proportional to the length of the time interval $T \cdot t_{clk}$ that the signal is allowed to propagate through the chain following

$$N \cdot t_p = T \cdot t_{clk} \Rightarrow T \leq \frac{N \cdot t_{p_{\min}}}{t_{\mathrm{clk}}}. \tag{5}$$

The t_{clk} is referred to the system clock while $t_{p_{\min}}$ presents the minimum propagation delay of a latch. The T factor has a lower limit proportional to the minimum propagation delay of the latch (fast propagation delay)and can be set using an FSM clocked by the system clock. A difficulty in the implementation of this topology is identified in the complexity of the mechanism which samples the output of the delay elements. This sampling logic requires a number of scan flip-flops equivalent to the number of the delay elements which are flip-flops with the additional ability of forming a shift register through which their values can be shifted out, hence an additional logic level comprising multiplexers is required. In Fig. 4 there is a chain of latches, flip-flops, multiplexers, an FSM implemented using FPGAs LUT and a counter implemented using FPGA DSP48. The flip-flops and the multiplexers implement a shift register used to capture the values of the latches and subsequently transfer them, in a serial way, to a counter which computes the number of delay elements through which the signal propagated. In this approach, there is only the input signal and the propagation signal passes

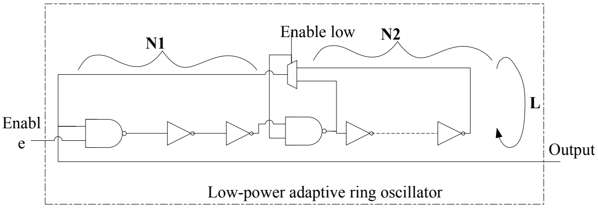


Fig. 5. Introduced low power adaptive ring oscillator

through the chain cannot be stalled. As a result, the flip-flops should be well synchronized in order to be able to capture the length of the signal propagation in a given time interval with accuracy.

The power consumption in the feed-forward approach is relatively high compared to the feedback loop organization. Since it is not possible to terminate the propagation of the signal though the delay chain, even if the propagating signal does not reach all elements to measure delay, it would eventually reach all stages of the chain at some point. Furthermore, power is consumed by the flip-flops as well. The process of resetting the flip-flop chain can be implemented by means of a reset signal. There is also power consumption at the multiplexers and at the counter as well. Finally there is the power consumption on the counter. The sum of all power dissipation components is

$$P_{\text{feed forward}} = N \cdot [P_{\text{delay}} + P_{\text{register}} + P_{\text{mux}}] + P_{\text{counter}} + P_{\text{FSM}} \tag{6}$$

Compared to the feedback loop topology, in this topology the power consumption of the main body of the delay sensor, i.e., the chain, is only a fraction of the total power consumption. The reduction of the power cannot be significant without compromising accuracy since a lot of the elements are responsible for the power consumption which cannot be removed.

4 Proposed Low-Power Delay Sensors

4.1 Feedback-Loop Low Power Design

To reduce power consumption in the feedback-loop architecture, the architecture presented in Fig. 5 is introduced. In this architecture, there is an additional mechanism providing the ability to enable or disable a part of the ring oscillator. Enabling the additional part, the produced frequency is reduced and thus the power consumption is only a fraction of the initial power consumption. In case of additional resolution is required, only the $N1$ part, as denoted to Fig. 5, is used and thus higher frequency is generated by the ring oscillator. This design combined with a dynamic change of the L factor during runtime is able to further reduce the power consumption. The total power consumption of the proposed organization can be described by

$$P_{tot} = a \cdot P_{N_1} \ + \ (1-a) \cdot P_{N_1+N_2} \tag{7}$$

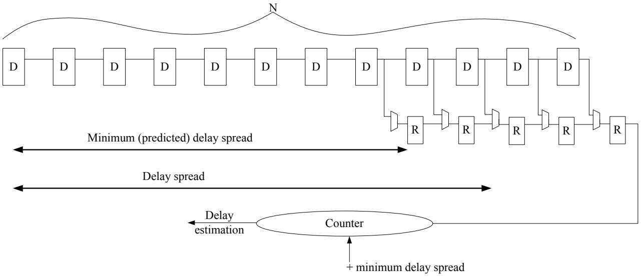


Fig. 6. Proposed low-power feed-forward delay sensor

where a denotes the percentage of the time where only the N_1-length part is active. Let β denote the ratio $\beta = \frac{P_{N_1}}{P_{N_1+N_2}}$. Then, power reduction γ due to (7) is computed as

$$\gamma = \frac{a \cdot P_{N_1} + (1-a) \cdot P_{N_1+N_2}}{P_{N_1+N_2}} = 1 - a + a \cdot \beta = 1 - a \cdot (1-\beta) \qquad (8)$$

As an example assuming $\beta = 0.7$ and $a = 0.3$, then $\gamma = 0.91$.

4.2 Feed Forward Low Power Design

One method to reduce both power and space requirements in the feed forward sensor is to reduce the number of the registers and multiplexers required in the capturing part. This can be achieved when it is possible to estimate the minimum and maximum length of the propagation of the signal in advance. This based on (5) which defines the T factor proportional to the N factor. Since the T factor is determined, the measured propagation can be calculated as

$$K_p = T \cdot \frac{t_{clk}}{t_p} \qquad (9)$$

thus, the number of the counted edges is upper and lower bounded. The upper limit equals to N while the lower limit is proportional to the slowest delay propagation

$$\frac{t_{clk}}{t_{p_{max}}} \leq K_p \leq N \qquad (10)$$

thus, if the maximum propagation delay of latches denoted as $t_{p_{max}}$ is known, it is possible to remove the unreachable registers from the scan chain. Such a design is depicted in Fig. 6. Since the minimum delay is estimated, the number of flip-flops is reduced by removing those that correspond to the shortest signal propagation anticipated. Due to circuit simplification power reduction is achieved.

5 Experimental Results

In this paper, we experimentally characterize the delay sensors by means of an extensive design space exploration. An experimental setup has been developed

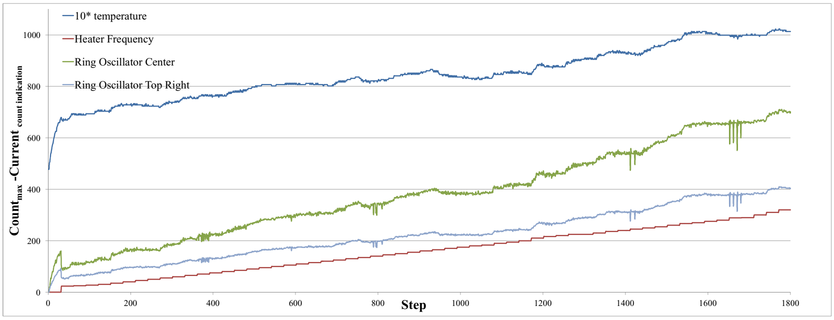


Fig. 7. Feedback loop sensitivity

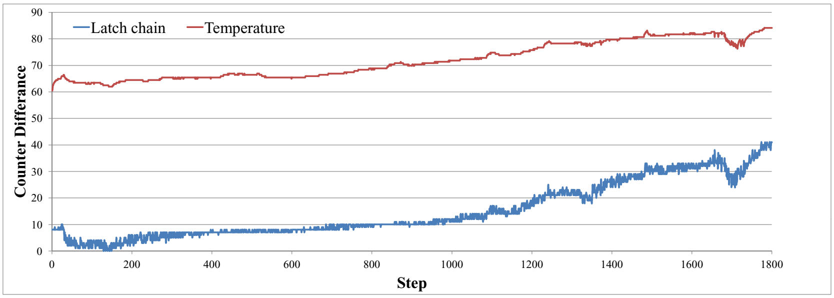


Fig. 8. Feed Forward Sensitivity

in hardware in order to emulate and control real time in-die conditions. To evaluate resolution, we use as a reference point the temperature sensors available on chip in the target FPGA, namely a Virtex-5 on the ML507 evaluation board [12] [10]. Several techniques have been reported in the corresponding literature [13] [6] to heat up an FPGA, such as the use of external heater or supply voltage manipulations. In this study, a design with specific characteristics is implemented as a heater. This design has the maximum switching activity and utilizes large numbers of low-complexity circuits. The controllability is sought through controlling the clock frequency of the heater. This can be achieved by using a Digital Clock Manager (DCM) [12] in dynamic reconfiguration mode. In this study every experiment lasts about two hours and 1800 measurements are taken. Heater clock frequency spans a range from 0 to 320 MHz in 60 steps. Several experiments took place in order to verify the delay sensing of the introduced architectures.

Figs. 7 and 8 depict the sensitivity of a resolution of the delay sensor compared to the indication of the on-chip temperature sensor. These results show the difference between the maximum possible value of the counter indication minus the corresponding counter indication. In Fig. 7 the sensitivity of a feedback loop architecture that uses a five-stage ring oscillator ($N = 5$) is presented while the L factor is set to 5000 time units referred to a 100 MHz clock. In Fig. 8 a latch

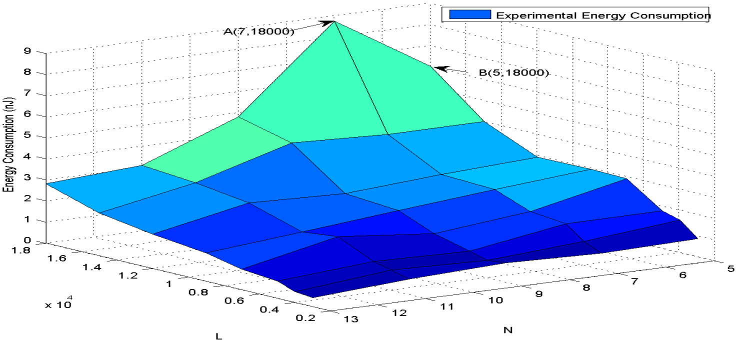


Fig. 9. Experimental Energy consumption

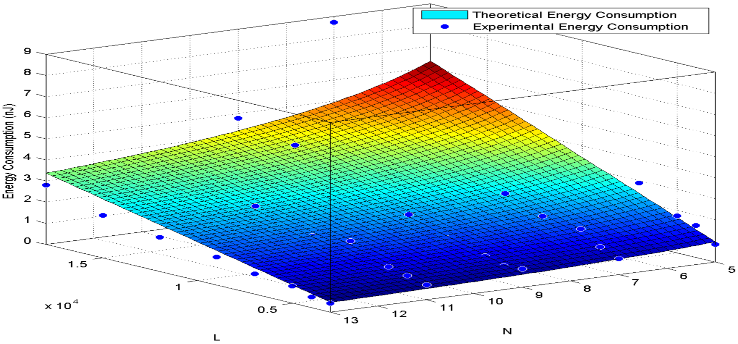


Fig. 10. Theoretical Energy consumption

chain of 2000 taps is used and the T factor has the value of 100 using the same reference clock.

Power consumption has been computed using the Xilinx XPower tool with annotation information from VCD files obtained from ModelSim. According to the results, the model of (1)–(3) is not able to describe the power consumption fully since the factor of routing and physical placement should more accurately be taken into consideration. Thus equations can only present an estimation of the power consumption and cannot be used to provide measure of power consumption to FPGA. In Fig. 9 the energy consumption of the feedback loop architecture is depicted in a three dimensional plot as a function of N and L. Fig. 10 depicts the theoretical energy of the feedback loop architecture. The theoretical estimation based on (3) where the various coefficients are determined by fitting to the experimental data. A similar analysis is presented in [14] where the corresponding frequency of a ring oscillator as a function of power voltage supply and temperature is presented. Here, we express energy as a function of the design parameters N, L. In Figs. 9 and 10 the x-axis is the number of stages in the ring oscillator N, the y-axis is the L factor and the z-axis is the energy demands counted in nJ (nano-Joules) per measurement. In Fig. 9, there are local minima of power consumption, thus if a specific resolution is required, a reference point with minima power consumption can be conduced.

By exploiting the exact shape of the energy surface, taking into consideration sensor resolution requirements, energy minimization can be achieved. In particular, options N and L are chosen for the ring oscillator architecture that allow requires resolution at low energy consumption. As an example, it is possible to reduce the energy by 31% given the desired resolution as depicted in Fig. 9, where N and L can be set to $(5, 18000)$ (point B) instead of $(7, 18000)$ (point A) to achieve a comparable resolution level.

6 Conclusion

This paper comparatively studies two topologies for all-digital delay sensors and the corresponding FPGA implementation issues. The two topologies are compared for power consumption and optimized topologies are introduced here aiming on low power consumption. To our knowledge, the obtained circuits and the corresponding design space exploration, are new. Given the detailed design space, it becomes possible to define the factors N and L in order to achieve the desired resolution with the minimum power consumption. An 31% reduction of power consumption is achieved while maintaining comparable resolution levels. In terms of resolution and power consumption the feed-forward topologies cannot keep up with the feedback loop designs. Furthermore, the optimized design presented in Fig. 5 can be used as a multi-resolution structure providing even lower levels of power consumption dynamically traded for resolution. The feedback-loop topologies offer a compact configurable design suitable for delay variation observation and combined with the proper computation mechanism capable of operating as thermal sensors. This paper presents a perspective of analysis of delay sensor structures which highlights the impact of the parameters on the power consumption and resolution.

Acknowledgement. We acknowledge the support of the END project from the ENIAC Joint Undertaking under grant agreement no. 120214 and from the Greek OPCE II (2007-2013) of the NSRF programme.

References

[1] Bowman, K., Tokunaga, C., Tschanz, J., Raychowdhury, A., Khellah, M., Geuskens, B., Lu, S., Aseron, P., Karnik, T., De, V.: All-digital circuit-level dynamic variation monitor for silicon debug and adaptive clock control. IEEE Transactions on Circuits and Systems - Part I: Regular Papers 58(9), 2017–2025 (2011)

[2] Rabaey, J.M., Chandrakasan, A., Nikolic, B.: Digital Integrated Circuits, 2nd edn. Prentice Hall (January 2003)

[3] Wu, L., Walker, D.: A fast algorithm for critical path tracing in VLSI digital circuits. In: 20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, DFT 2005, pp. 178–186 (October 2005)

[4] Lucas, G., Dong, C., Chen, D.: Variation-Aware placement with Multi-Cycle statistical timing analysis for FPGAs. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 29(11), 1818–1822 (2010)

[5] Zhang, Y., Srivastava, A.: Accurate temperature estimation using noisy thermal sensors. In: 46th ACM/IEEE Design Automation Conference, DAC 2009, pp. 472–477 (July 2009)

[6] Franco, J., Boemo, E., Castillo, E., Parrilla, L.: Ring oscillators as thermal sensors in FPGAs: experiments in low voltage. In: Programmable Logic Conference (SPL), 2010 VI Southern, pp. 133–137 (March 2010)

[7] Parhi, K.K.: VLSI Digital Signal Processing Systems: Design and Implementation, 1st edn. Wiley-Interscience (January 1999)

[8] Datta, B., Burleson, W.: Low-power and robust on-chip thermal sensing using differential ring oscillators. In: 50th Midwest Symposium on Circuits and Systems, MWSCAS 2007, pp. 29–32 (August 2007)

[9] Balankutty, A., Chih, T., Chen, C., Kinget, P.: Mismatch characterization of ring oscillators. In: Custom Integrated Circuits Conference, CICC 2007, pp. 515–518. IEEE (September 2007)

[10] Xilinx: DS202: Virtex-5 data sheet: DC and switching characteristics (October 2006)

[11] Agarwal, K.: On-die sensors for measuring process and environmental variations in integrated circuits. In: Proceedings of the 20th Symposium on Great Lakes Symposium on VLSI, GLSVLSI 2010, pp. 147–150. ACM, New York (2010)

[12] Xilinx: UG190: Virtex-5 user guide (July 2006)

[13] Zick, K.M., Hayes, J.P.: On-line sensing for healthier FPGA systems. In: Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays, FPGA 2010, pp. 239–248. ACM, New York (2010)

[14] Orshansky, M., Nassif, S., Boning, D.: Design for Manufacturability and Statistical Design: A Constructive Approach, 1st edn. Springer Publishing Company, Incorporated (2010)

Observability Conditions and Automatic Operand-Isolation in High-Throughput Asynchronous Pipelines

Arash Saifhashemi and Peter A. Beerel*

University of Southern California
{saifhash,pabeerel}@usc.edu
http://async.usc.edu

Abstract. In this paper, we model conditional communication primitives of asynchronous circuits as three-valued logic operators and adopt the theory of observability don't cares to create a theoretical framework that can be used to guide the optimization of conditional communication in asynchronous circuits. In particular, using this framework we demonstrate how operand-isolation cells introduced by standard synthesis algorithms can guide the addition of conditional communication primitives to surround blocks of asynchronous logic with conditional communication reducing switching activity and power. Our experimental results show for a 32-bit ALU, we achieve an average of 53% power reduction for about a 4% increase in area with no impact in performance.

Keywords: Asynchronous Circuits, Observability, Operand-Isolation, Three-Valued Logic, Conditional Communication.

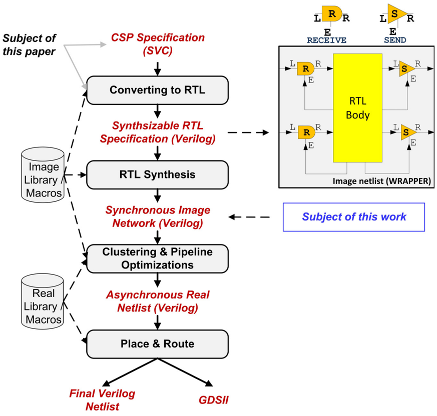
1 Introduction

In asynchronous circuits, carefully designed *conditional communication* on channels allow sub-circuits to be active *only* when necessary computation is performed [1]. Some asynchronous flows translate the conditionality that is explicit in the high-level specification into conditional communication [2]. Others translate clock-gating structures derived from an RTL description to conditional asynchronous split-merge architectures [3,4]. Our focus is a recently commercialized ASIC flow for high-performance asynchronous circuits called Proteus [5] that uses commercial synchronous synthesis and place-and-route tools.

In order to improve power-efficiency, Proteus relies on the designer to *manually* decompose the high-level specification and introduce conditional communication. Manual decomposition is tedious and error-prone. This paper provides a theoretical framework that allows automation of this task and moreover enables using commercial synchronous power-optimization tools and the adoption of common power-optimization techniques, such as operand-isolation [6] and clock-gating which are largely based on observability don't cares (ODCs) [7]. The contributions of this paper are as follows:

* Peter A. Beerel is also Chief Scientist, Switch and Router Division, Intel Corporation.

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 205–214, 2013.



(a) Proteus Flow

```
module ALU( e1of2_4.In  OP,
   e1of2_32.In  I1,I2
   e1of2_32.Out O);
 logic [4-1:0]  op;
 logic [32-1:0] i1, i2, o;
 always begin
  forever begin
    I1.Receive(i1); I2.Receive(i2);
    OP.Receive(op);
   unique case (op)
    4'b0001: o = i1 & i2 ;
    4'b0010: o = i1 + i2 ;
    4'b0100: o = i1 - i2 ;
    4'b1000: o = i1 * i2 ;
   endcase
   O.Send(o);
  end //forever
 end //always
endmodule
```

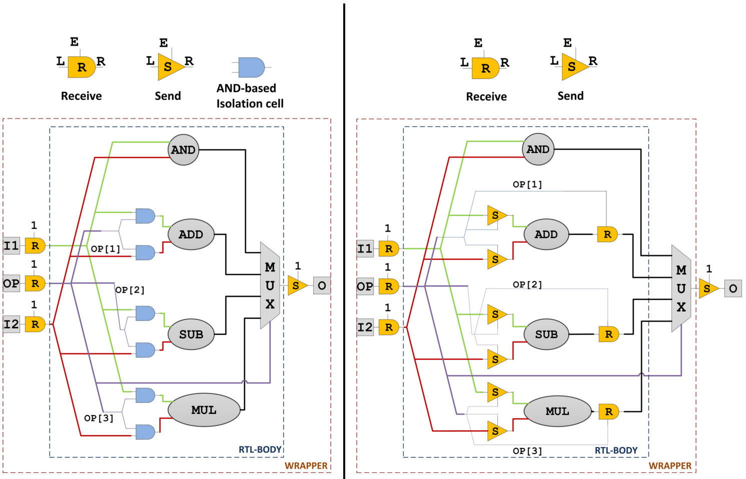
(b) ALU in SystemVerilogCSP

Fig. 1. A 32-bit ALU and its WRAPPER with isolation cells

- The single-rail synchronous model of asynchronous netlists in Proteus is enhanced and formalized by adopting a *three-valued* logic system to model the absence of communication on channels. The theory of *observability don't care* is applied to this system which allows adoption and reasoning about mature synchronous power optimization techniques for asynchronous systems.
- A novel application of this model for automatic implementation of operand-isolation using synchronous CAD tools is presented. Moreover, a cost function for pre-layout evaluation of the addition of isolation cells is provided.

2 Background and Motivation

Proteus [5] is a commercially-proven asynchronous ASIC flow illustrated in Figure 1a. It supports a class of *communicating sequential processes* (CSP) [8] that consist of an *initialization* block followed by a *forever* loop. In this flow, the first step is to convert the high-level specification, described in *SystemVerilogCSP* (SVC) [9], to a synthesizable single-rail RTL netlist called WRAPPER. As an example, the SVC description and the corresponding WRAPPER for a 32-bit ALU are illustrated in Figure 1b and 2a. The WRAPPER instantiates special RECEIVE/SEND library cells to model conditional communication and a synthesizable RTL-BODY that implements the core logic of the process. In this model, each iteration of the *forever* loop in the SVC description is mapped into one clock cycle of RTL-BODY. In the case of conditional inputs, the RTL-BODY *asserts* the condition of the communication on each RECEIVE cell's E input. Only if the condition is *1*, does it *consume* the RECEIVE cell's output port R. RTL-BODY then calculates and *asserts* the output data and output condition



(a) WRAPPER with AND-based isolation cells

(b) Replacing AND gates with conditional communication

Fig. 2. A 32-bit ALU before (left) and after (right) proposed optimization

values on each SEND cell's E and L ports respectively. The data is *asserted* on the SEND cell's R only if its E is *1*. Currently, the exact behavior of RECEIVE/SEND cells in this single-rail model is not defined. This paper proposes a formal definition based on *three-valued* logic in Section 3.

By delegating conditionality to stand-alone library cells, the RTL-BODY becomes a synthesizable unconditional module, i.e., in each clock cycle *all* of its inputs are available and *all* of its outputs are asserted. The synthesis tool recognizes RECEIVE and SEND cells as hard macros and only synthesizes the RTL-BODY into a netlist of *image* library cells, i.e. single-rail cells that do not physically exist, but are replaced by their real asynchronous dual-rail/1-of-N equivalents after synthesis. These real cells are then clustered into asynchronous fine-grained PCHB [10] pipeline stages. The RECEIVE/SEND cells are also replaced by their real asynchronous dual-rail/1-of-N equivalents. Due to the fine-grained nature of PCHB template, clustering yields a complex non-linear pipeline. Asynchronous pipeline optimization algorithms [11] are then applied to ensure meeting the target throughput. Finally, the asynchronous netlist is given to a commercial place-and-route tool for physical design.

It is important to note that the Proteus flow supports conditional communication only at the boundary of SVC descriptions using RECEIVE/SEND cells described in Figure 3. When the value of the *token* [1] on channel E of a real RECEIVE cell is *0*, no token is received on its input channel L. Similarly, if the value of the token on channel E of a SEND cell has value *0*, no token will be sent on its R channel, and hence no switching at its output. On the other hand, the internals of each CSP process (modeled by RTL-BODY) is implemented with *unconditional* asynchronous cells, i.e., every real library cell implementing RTL-BODY expects a token on *all* of its input channels before generating its output

```
always begin
    E.Receive(e);
    if (e==1) L.Receive(d);
    else    d=0; //dummy value
    R.Send(d);
end
```

(a) RECEIVE

```
always begin
    E.Receive(e);
    L.Receive(d);
    if (e==1)
      R.Send(d);
end
```

(b) SEND

Fig. 3. SVC description of real RECEIVE and SEND cells

token. Therefore, if the value of the token on E is *0*, RECEIVE still sends a *dummy* token with value *0*. Similarly, when the token on channel E of a SEND cell is *0*, this cell still expects a token coming from a real asynchronous cell on its L channel although it does not generate any output token on its R channel.

2.1 A Motivating Example

Operand-isolation is the direct application of observability don't cares, where propagation of switching activity through mathematical operators is blocked by placing isolation cells (such as AND gates) at inputs of operators whose outputs are often not observable [6]. Consider the 32-bit ALU described[1] in SVC in Figure 1b. The synthesized WRAPPER in which AND-based isolation cells are automatically inserted by the RTL synthesis tool is shown in Figure 2a. Notice that since in this example all inputs and outputs are unconditional, the enable inputs of RECEIVE/SEND cells are connected to *1*.

Although AND-based isolation cells are effective in synchronous design, they will not block the propagation of switching activity in the resulting asynchronous netlist. In particular, after replacing the image cells with their asynchronous counterparts, the AND cells associated with an isolated operator still unconditionally send the value *0* to down-stream stages. Moreover, unlike in synchronous netlists, using dual-rail/1-of-N handshaking forces switching activity on data wires even if the same value is sent on a channel over and over.

Our proposed solution, illustrated in Figure 2b, is to automatically and optimally translate these isolation cells into additional SEND/RECEIVE cells that truly isolate the operands by introducing additional conditional communication into the circuit and thus reduce switching activity. We use three-valued logic model to prove the correctness of our approach.

3 Three-Valued Logic Observability Conditioning

Three-valued logic (3VL) has been used for synthesizing asynchronous systems in a variety of forms (e.g. [13]). In this section, we adopt a 3VL model to introduce

[1] In SVC, M 1-of-N channels are modeled by an *interface* [12] called *e1of_N_M*. The N in 1-of-N channels should not be confused with the proposed N value in Sec. 3.

and optimize conditional communication in a circuit. In particular, we model the image netlist of Proteus as a network of 3V gates, where the RECEIVE and SEND cells are also modeled as 3V functions.

Each variable can take one of the three values in the set $\mathcal{T} = \{\mathit{0}, \mathit{1}, N\}$. Values *0* and *1* are the common two Boolean logic values. The value N models the condition of *no communication* (a.k.a a *spacer* [1]) on a channel and should not be confused by the traditional concept of a *don't care*. Variables are updated based on a set of 3V tables as exemplified in Figure 4. The $\wedge$ and $\vee$ operators behave similar to Boolean AND and OR operators as long as none of the inputs is N, otherwise the output becomes N. By definition, the inverting operator $\neg$ (not shown) behaves like the Boolean inverter when its input is *0* or *1*, but its output is N, when its input is N. The output of the equivalence operator $\equiv$ always has a value of *0* or *1*. The RECEIVE operator (denoted as $\circledR$) behaves like a buffer when its enable input E is *1*. If E is *0*, the output is always a dummy value *0*. The SEND operator (denoted as $\circledS$) also behaves like a buffer when E is *1*. However, the SEND's output is N when E is *0* or N.

		A		
	$\wedge$	**0**	**1**	**N**
B	**0**	0	0	N
	1	0	1	N
	N	N	N	N

(a) $A \wedge B$ (Logical AND)

		A		
	$\vee$	**0**	**1**	**N**
B	**0**	0	1	N
	1	1	1	N
	N	N	N	N

(b) $A \vee B$ (Logical OR)

		A		
	$\equiv$	**0**	**1**	**N**
B	**0**	1	0	0
	1	0	1	0
	N	0	0	1

(c) $A \equiv B$ (Equivalence)

		E		
	$\circledR$	**0**	**1**	**N**
L	**0**	0	0	N
	1	0	1	N
	N	0	N	N

(d) $L \circledR E$ (RECEIVE)

		E		
	$\circledS$	**0**	**1**	**N**
L	**0**	N	0	N
	1	N	1	N
	N	N	N	N

(e) $L \circledS E$ (SEND)

Fig. 4. Operators in three-value logic

A 3V literal [14] $x_i^{c_i}$ is a 2V function of the form:

$$x_i^{c_i} : \mathcal{T} \longrightarrow \mathcal{B} \quad , \quad x_i^{c_i} = (x_i \equiv \gamma_1) \vee (x_i \equiv \gamma_2) \vee \ldots (x_i \equiv \gamma_k),$$

where $\gamma_j \in c_i \subseteq \mathcal{T}$, and $\mathcal{B} = \{\mathit{0}, \mathit{1}\}$. For example, $x^{\{\mathit{0},N\}}$ is *1* when x is *0* or N.

We use the term *unconditional function* to distinguish between logic functions and SEND/RECEIVE operators $\circledS$ and $\circledR$. In particular, a function that can be obtained only by composition of $\wedge$, $\vee$, and $\neg$ is called an unconditional function. A *combinational 3V network* is a directed graph *G(V,E)*, where V is the set of nodes and E is the set of edges, in which each node represents a 3V function with a single 3V output and several 3V inputs. Moreover, there is a directed edge $(u, v) \in E$ from node u to node v, if a function represented by node v explicitly depends on the output variable at node u. A node v is *unconditional* if the function represented by v is unconditional. The output of a 3V unconditional function is N if and only if at least one of its inputs is N:

Lemma 1. *Let* $y = f(x_1, x_2, \ldots, x_n)$ *be an unconditional 3V function:*

$$\exists x_i = N \longleftrightarrow (y = N) \quad (1 \leq i \leq n)$$

Proof: Direct corollary of the tables of Figure 4 and the definition of unconditional functions. ■

The next lemma states the right distributivity of a SEND function over an unconditional function. Using this lemma, one can move a SEND node from the output of an unconditional node to its inputs:

Lemma 2 (SEND Reconditioning). *Let $y = f(\mathbf{x}) = f(x_1, x_2, \dots, x_n)$ be an unconditional 3V function, and $e \notin TFO(y)$, then*[2]*:*

$$f(x_1, x_2, \dots, x_n)\circledS e = f(x_1\circledS e, x_2\circledS e, \dots, x_n\circledS e)$$

Proof (by case analysis on e):

- $e = 0 : f(\mathbf{x})\circledS 0 = N \overset{Lem.1}{=} f(N, \dots, N) = f(x_1\circledS 0, x_2\circledS 0, \dots, x_n\circledS 0)$
- $e = 1 : f(\mathbf{x})\circledS 1 = f(\mathbf{x}) = f(x_1\circledS 1, x_2\circledS 1, \dots, x_n\circledS 1)$
- $e = N : f(\mathbf{x})\circledS N = N \overset{Lem.1}{=} f(N, \dots, N) = f(x_1\circledS N, x_2\circledS N, \dots, x_n\circledS N)$

■

In a 3V network, we call two variables x_1 and x_2 *synchronized*, denoted by $x_1 \textcircled{c} x_2$, if the following expression is *always 1*:

$$(x_1^{\{0,1\}} x_2^{\{0,1\}}) \vee (x_1^{\{N\}} x_2^{\{N\}})$$

3.1 Observability Condition and Operand Isolation

In a 3VL network, the *local observability partial care* of input x_j of a node representing a function f is denoted as $OPC(f, C, x_j)$, where $C \subseteq \mathcal{T}$ is the set of values that x_j can take without affecting the value of f. OPC can be computed using the following equation [15]:

$$OPC(f, C, x_j) = f^0_{x_j^{c_1}} f^0_{x_j^{c_2}} \dots f^0_{x_j^{c_n}} + f^1_{x_j^{c_1}} f^1_{x_j^{c_2}} \dots f^1_{x_j^{c_n}} + f^N_{x_j^{c_1}} f^N_{x_j^{c_2}} \dots f^N_{x_j^{c_n}}, \tag{1}$$

where $f^{\{l\}} : \mathcal{T}^n \rightarrow \mathcal{B}$ is a 2V function ($l \in \mathcal{T}$) of n 3V variables that defines the set of minterms in $\mathcal{T}^n$ in which the value of f is l. Function $f^l_{x_j^k}$, $k \in \mathcal{T}$ is the *cofactor* of $f^{\{l\}}$ with respect to literal $x_j^{\{k\}}$. It is independent of x_j and preserves the onset of $f^{\{l\}}$ whenever $x_j = k$.

The global observability partial care of a variable x in a 3V network, denoted by $GOPC(C, x)$, is the condition under which the value of no output variables of the network is affected as the value of x changes in the range $C \subset \mathcal{T}$. It is not hard to show:

$$OPC(f, C, x) = 1 \longrightarrow GOPC(C, x) = 1$$

The general idea of this paper is that if we know the output of a node is not observable, we can transform the network into an equivalent network which may consume less power. This is formalized as follows.

[2] We let TFO(e) represent the transitive fanout of node e.

Lemma 3. *Let u be an unconditional node with inputs $(x_1, x_2, \ldots, x_n)$ and output y_1, implementing function $y_1 = f(\mathbf{x})$. For a variable $e \notin TFO(y_1)$, if $e^{\{0\}}$ implies $GOPC(\mathcal{B}, y_1) = 1$ and if $\forall x_i \in \{x_1, \ldots, x_n\} : e \textcircled{c} x_i$, replacing u with u_c implementing function $y_2 = \big(f(\mathbf{x}) \circledS e\big) \circledR e$ preserves equivalence.*

Proof (by case analysis on e): In the case $e^{\{0\}}$, based on the assumption, it is implied that $GOPC(\mathcal{B}, y_1) = 1$, therefore replacing $y_1 = f(\mathbf{x})$ with $y_2 = \big(f(\mathbf{x}) \circledS e\big) \circledR e$ does not change the value of any primary output variable if we show that when y_1 takes values in $\mathcal{B}$, y_2 also only takes values in $\mathcal{B}$:

$$\forall x_i \in \{x_1, \ldots, x_n\} : e \textcircled{c} x_i \overset{Lem. 1}{\longrightarrow} y_1 \textcircled{c} e$$

Therefore:

$$y_1^{\mathcal{B}} \overset{y_1 \textcircled{c} e}{\longrightarrow} e^{\mathcal{B}} \xrightarrow{Fig.\ 4d,\ 4e} y_2^{\mathcal{B}}$$

For the other two cases, we prove: $y_2 = \big(f(\mathbf{x}) \circledS e\big) \circledR e = f(\mathbf{x}) = y_1$:

- $e^{\{1\}} : y_2 = \big(f(\mathbf{x}) \circledS 1\big) \circledR 1 = f(\mathbf{x}) \circledR 1 = f(\mathbf{x}) = y_1$
- $e^{\{N\}} : y_2 = \big(f(\mathbf{x}) \circledS N\big) \circledR N = N \circledR N = N \overset{Lem. 1}{=} f(N, \ldots, N) = y_1$

■

Lemma 3 states that if the output of a node u is not observable, one can add a SEND followed by a RECEIVE at the output of u. Using Lemma 2, we can move the SEND to the inputs:

Theorem 1 (GOPC Conditioning) *Let u be an unconditional node with inputs $(x_1, x_2, \ldots, x_n)$ and output y_1, implementing function $y_1 = f(\mathbf{x})$. For a variable $e : e \notin TFO(y_1)$, if $e^{\{0\}}$ implies $GOPC(\mathcal{B}, y_1) = 1$, and $\forall x_i \in \{x_1, \ldots, x_n\} : e \textcircled{c} x_i$, replacing u with u_c implementing function $f(x_1 \circledS e, \ldots, x_n \circledS e) \circledR e$ preserves equivalence.*

Proof (by case analysis on e): In the case $e^{\{0\}}$, the proof is similar to Lemma 3. For the other two cases we prove $y_1 = y_2$:

$$y_1 = f(\mathbf{x}) \overset{Lem. 3}{=} \big(f(\mathbf{x}) \circledS e\big) \circledR e \overset{Lem. 2}{=} f(x_1 \circledS e, \ldots, x_n \circledS e) \circledR e = y_2$$

■

Theorem 1 says that knowing the observability of a function $y = f(x_1, ..., x_n)$, one can isolate f by placing a SEND before all of its inputs and a RECEIVE after its output as shown in Figure 2b without affecting the values of primary outputs. In particular, we enable operand-isolation in the synchronous RTL-synthesis tool and recognize that when the enable input e of the isolation cells is 0, the output of the corresponding operator is not observable. When e evaluates to 1, just like AND-based isolation cells of Figure 2a, the values of the primary outputs are unchanged. When e evaluates to 0, the SEND cells generate an N value at the inputs of the operator and the RECEIVE cells generate a 0 at the output of the operator but based on Theorem 1, since the output is not observable under this condition, the primary output values remain unaffected.

Note that that RECEIVE cells are necessary to prevent the propagation of value N generated by SEND cells associated with isolated operators through the logic gates after the isolated operators.

3.2 Pre-layout Cost and Benefit Evaluation

The RTL synthesizer is constrained to add operand isolation only on non-critical paths. It is desired to decide whether the extra switching power and area associated with adding RECEIVE and SEND cells is justified by the amount of saved power *before* physical design. Here we present a pre-layout cost and benefit estimation function to evaluate the cost and benefit of isolating each operator and only commit the ones whose costs are justified by their benefits.

Let V be the set of all nodes in a given network and $V_f \subset V$ be the set of gates implementing a function $f(x_1, ..., x_n)$. Let W_u be the switching power of each gate $u \in V$ when it is active, i.e. the amount of switching power necessary for communicating on all input channels, calculating the output, and communicating on the output channel. The total switching power of the circuit is P^{total}, and the switching power consumption in f without operand-isolation is P_f^o:

$$P^{total} = \sum_{u \in V} W_u \quad , \quad P_f^o = \sum_{u \in V_f} W_u \tag{2}$$

Assuming we know the *activity ratio* r_f of each operator f (the ratio of the iterations that f is executing useful operations to all iterations), the switching power of f after isolation, P_f^i, is:

$$P_f^i = r_f \sum_{u \in V_f} W_u + K(n+1) \quad , \quad 0 \le r_f \le 1 \tag{3}$$

where the second term accounts for switching activity of n isolating SEND cells at inputs and one RECEIVE cell at the output. For simplicity, we assumed the switching power for RECEIVE/SEND cells is K. We define the relative benefit B_f of isolating operator f to be:

$$B_f = \frac{\Delta P_f}{P_{total}} = \frac{P_f^o - P_f^i}{P_{total}} \tag{4}$$

The area cost of isolating f with n inputs and one output is $Cost_f = L(n+1)$, where L is the area cost of a SEND or a RECEIVE cell.

3.3 Experimental Results

This section presents a case-study of an ALU, a classic example of where operand-isolation can be beneficial. Assuming $L = 1$, the total pre-layout cell area estimation after synthesis is 1516 units. The area cost of isolating each operator is 96 units: 64 SEND and 32 RECEIVE cells. To estimate the switching power reduction, we used Equation 4 and a simple model in which the switching power of all cells are normalized and equal to *1* unit, i.e., $\forall u \in V : W_u = 1$, and $K = 1$. Using Equation 4 with a uniform $r_f = 0.25$ for each operation, we get $B_{ADD}, B_{SUB} \approx 0\%$ and $B_{MUL} = 53\%$, which suggests that the relative benefit of isolating ADD and SUB is negligible, whereas the benefit of isolating MUL can be significant.

Next, we implemented the proposed operand-isolation flow within Proteus and used a commercial power analysis tool to estimate the post-layout power values of the final asynchronous netlist. We used a proprietary TSMC 65nm PCHB-based cell library for which unfortunately internal power was not available and thus ignored. To measure switching power, we used a Value Change Dump [12] file generated from simulating the post-layout asynchronous netlist at maximum throughput (1.1GHz) with different mixes of op-codes.

Table 1. Post-layout total switching power measurements (mW).
$\boldsymbol{P^o}$*: No operand isolation.* $\boldsymbol{P^i_M}$*: only MUL is isolated.*
$\boldsymbol{P^i_{ASM}}$*: ADD, SUB, and MUL are isolated.* $\boldsymbol{P^m}$*: manual decomposition*

Activity	$\boldsymbol{P^o}$	$\boldsymbol{P^i_M}$	$\boldsymbol{P^i_{ASM}}$	$\boldsymbol{P^m}$	$\frac{(P^o - P^i_M)}{P^o}$	$\frac{(P^o - P^i_{ASM})}{P^o}$	$\frac{(P^m - P^i_{ASM})}{P^m}$
$\boldsymbol{r_{\mathrm{AND}} = 1}$	110	34	34	39	69%	69%	13%
$\boldsymbol{r_{\mathrm{ADD}} = 1}$	110	34	38	41	69%	66%	8%
$\boldsymbol{r_{\mathrm{SUB}} = 1}$	110	34	38	42	69%	65%	9%
$\boldsymbol{r_{\mathrm{MUL}} = 1}$	110	106	106	108	4%	3%	2%
$\boldsymbol{r_{f_i} = 0.25}$	110	52	54	56	53%	51%	3%

The results are shown in Table 1. In rows *2* to *5*, the ALU only executed one type of operation. The last row shows the results when the ALU performed random but uniformly distributed operations, i.e., for each operator f, $r_f = 0.25$. In this setting, if we only isolate MUL (P^i_M), the total switching power reduces by *53*%. The fourth column (P^i_{ASM}) shows that if we simultaneously isolate ADD, SUB, and MUL, the power savings drops to *51*%, which is consistent with our simple pre-layout analysis. Lastly, the last column shows the power savings of our proposed method is better than those of a design in which the ADD, SUB, and MUL are manually isolated via CSP decomposition. This suggests that not only is the automatic approach efficient, but that manual decomposition suffers from the fact that the synthesis tool cannot optimize across CSP boundaries.

The post-layout area cost of only isolating MUL is *4*% whereas if we simultaneously isolate ADD, SUB, and MUL, the cost increases to *13*%. In addition, the area cost of our proposed method is lower than that of manual decomposition by *8*%, further indication of the efficiency of our approach.

Finally, it is important to note that the throughput of the circuits was not impacted by the introduction of the SEND/RECEIVE, as the pipeline optimization steps within Proteus automatically compensated for their existence. Moreover, the RTL synthesizer was constrained to add operand isolation only on non-critical paths.

4 Summary and Conclusions

We presented an automatic method to reduce the power consumption of high-performance asynchronous ASICs using operand isolation. We showed that in a classic ALU executing uniformly distributed operations, a *53%* power saving can be achieved for only *4%* area cost with no impact on performance. Our results are comparable to what is achievable using manual decomposition.

While three-valued logic has been used for synthesizing asynchronous circuits in the past, we believe it is also a suitable framework for optimization of conditional communication and can be used to formalize a range of other optimization techniques, including translating synthesized clock-gating structures into conditional communication and optimizing the location of the boundary between communication and computation cells.

References

1. Beerel, P.A., Ozdag, R., Ferretti, M.: A Designer's Guide to Asynchronous VLSI. Cambridge University Press (2010)
2. Wong, C.G., Martin, A.J.: High-level synthesis of asynchronous systems by data-driven decomposition. In: DAC 2003, pp. 508–513 (2003)
3. Manohar, R.: Systems and methods for performing automated conversion of representations of synchronous circuit designs to and from representations of asynchronous circuit designs. Patent 2007/0 256 038 A1 (2007)
4. Smirnov, A., Taubin, A.: Synthesizing Asynchronous Micropipelines with Design Compiler. In: SNUG Boston 2006 (2006)
5. Beerel, P.A., Dimou, G., Lines, A.: Proteus: An ASIC Flow for GHz Asynchronous Designs. IEEE D.&T. of Computers 28(5), 36–51 (2011)
6. Correale Jr., A.: Overview of the power minimization techniques employed in the IBM PowerPC 4xx embedded controllers. In: International Symposium on Low Power Design, pp. 75–80 (1995)
7. Bartlett, K.A., Brayton, R.K., Hachtel, G.D., Jacoby, R.M., Morrison, C.R., Rudell, R.L., Sangiovanni-Vincentelli, A., Wang, A.: Multi-level logic minimization using implicit don't cares. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 7(6), 723–740 (1988)
8. Hoare, C.: Communicating Sequential Processes. Prentice Hall (1985)
9. Saifhashemi, A., Beerel, P.A.: SystemVerilogCSP: Modeling Digital Asynchronous Circuits Using SystemVerilog Interfaces. In: CPA-2011: WoTUG-33, pp. 287–302. IOS Press (2011)
10. Lines, A.M.: Pipelined asynchronous circuits. California Institute of Technology. Tech. Rep, (revised 1995)
11. Beerel, P.A., Lines, A., Davies, M., Kim, N.H.: Slack matching asynchronous designs. In: ASYNC 2006, pp. 184–194 (2006)
12. IEEE Standard for SystemVerilog - Unified Hardware Design, Specification, and Verification Language, Std. (2009)
13. Wojcik, A.S., Fang, K.-Y.: On the Design of Three-Valued Asynchronous Modules. IEEE Trans. Comput. 29(10), 889–898 (1980)
14. Brayton, R.K., Khatri, S.P.: Multi-valued logic synthesis. In: Proceedingsof the Twelfth International Conference on VLSI Design 1999, pp. 196–205 (1999)
15. Yunjian, J., Brayton, R.K.: Don't cares and multi-valued logic network minimization. In: ICCAD 2000, pp. 520–525 (2000)

Dynamic Power Management of a Computer with Self Power-Managed Components

Maryam Triki [1,*], Yanzhi Wang[2], Ahmed C. Ammari[1,3], and Massoud Pedram[2]

[1] MMA Laboratory, National Institute of the Applied Sciences and Technology (INSAT),
Carthage University, Tunis, Tunisia
maryam.triki@gmail.com,chiheb.ammari@insat.rnu.tn
[2] Department of Electrical Engineering, University of Southern California,
Los Angeles, CA, USA
{yanzhiwa,pedram}@usc.edu
[3] Department of Elec. & Computer Engineering, Faculty of Engineering,
King Abdulaziz University, Jeddah, Saudi Arabia

Abstract. This paper presents a Dynamic Power Management (DPM) framework based on reinforcement learning (RL) technique which aims to save power in an Energy-Managed Computer (EMC) system with self power-managed components. The proposed online adaptive DPM technique consists of two layers: component-level and system–level global power manager (GPM). The component-level PM policy is pre-specified and fixed whereas the system-level global PM employs temporal difference learning on Semi-Markov Decision Process (SMDP) for model-free RL, and it is specifically optimized for a multi-type application framework. Experiments show that that the proposed HPM scheme enhances power savings considerably while maintaining a good performance level. In comparison with other reference systems, the proposed RL DPM approach performs well under various workloads, can simultaneously consider power and performance and achieves a wide and deep power-performance tradeoff curves.

Keywords: Dynamic Power Management (DPM), Reinforcement Learning (RL), Power optimization.

1 Introduction

Dynamic power management (DPM) refers to the selective shut-off or slow-down of system components that are idle or underutilized. Such technique has proven to be a particularly effective way of reducing power dissipation at system level [1]. An effective DPM policy should minimize power consumption while maintaining performance degradation to an acceptable level. The DPM methods proposed in the literature can be broadly classified into three categories: heuristic, stochastic, and learning based methods. The heuristic methods are based on the idea of predicting the length of idle time based on the computation history, and then shuts the device down if the predicted length of idle time justifies the cost. A decision to sleep will be made if the

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 215–224, 2013.

prediction indicates the idle period is longer than a specific value (the break-even time Tbe). Among these methods Srivastava et al. [2] use a regression function to predict the idle period length while Hwang et al. [3] propose an exponential-weighting average function for predicting the idle period length. Such techniques perform well only when the requests are highly correlated and they do not take performance constraints into account. The stochastic approaches can take into account both power and performance and are able to derive provably optimal DPM policies by modeling the request arrival times and device service times as stationary stochastic processes such as Markov Decision Processes (MDP) that satisfy certain probability distributions [4], [5], [6], [7]. The essential shortcoming of these methods is the need of an exact knowledge of the state transition probability function of the MDP. However, the workload of a complex system is usually unpredictable and its variations has a significant impact on the system performance and power consumption. Several recent works use machine learning for adaptive policy optimization [8, [9], [10, [11]. Compared to heuristic policies, machine learning-based methods such as Reinforcement learning can simultaneously consider power and performance, and perform well under various workload conditions.

All of the above-mentioned DPM works have focused on developing local component-level policies. However, a number of built-in power-management solutions already incorporated into various standards and protocols cannot be changed because they ensure the correct functionality of a device running the related protocol. In this sense, we consider such a device as an uncontrollable or self power-managed component.

A few research results related to the dynamic power management have been reported. Reference [16] uses a similar system set-up as this paper, to resolve the DPM of a Computer with self power-managed components. This approach is based on optimal policies that are computed offline using Continuous Time Markov Decision process (CTMDP) model of the system. In this approach, the request inter-arrival times and system service times are modeled as stationary processes that satisfy exponential probability distributions which is not always realistic given that such parameters are usually unpredictable. Reference [17] proposes a hierarchical adaptive DPM, where the term "hierarchical" refers to the manner in which the authors formulate the DPM policy optimization as a problem of seeking an optimal rule that switches policies among a set of pre-computed ones.

In this paper, we develop a novel online adaptive approach for Dynamic Power Management (DPM) of a computer with self power-managed components. The proposed approach consists of two layers: component-level and system–level global power manager (GPM). The component-level PM policy is pre-specified and fixed whereas the system-level global PM perform power management in a continuous-time and event-driven manner using temporal difference learning on Semi-Markov Decision Process (SMDP) for model-free Reinforcement Learning (RL) techniques. Using such RL approach, we do not assume that the characteristics of the power state transition are known, and we do not need to evaluate only one predefined policy, but to simultaneously learn the optimal policy and use that policy to control. For the proposed approach, the tradeoff between the power consumption and the performance (latency) can be controlled by a user defined parameter while reference [16] sets out to meet given performance constraints. We will also extend the work of reference [16] by considering non-stationary workloads: the optimal policy is learned under non-stationary workloads.

The remainder of this paper is organized as follows. Section 2 presents the DPM framework. Details of the proposed dynamic power management algorithm are given in Section 3. The experimental results are presented in Section 4, and we conclude our findings in Section 5.

2 DPM Framework

This paper focuses on reducing energy consumption of a computer system with built-in component-level local power managers. Basically, we propose an indirect approach that regulates the requests service flow of the self power managed components. Timeout policies are considered to be used for these local self power managed components. Under these policies, a Local Power Manager (LPM) directly controls the state transitions of the device. At the same time, the system-level Global Power Manager (GPM) helps the local power manager improve power efficiency by performing service flow regulation.

2.1 Model of the Service Request and the Service Provider

The workload source (SR), in general, has a non-stationary behavior.

For the sake of simplicity, let us consider a service provider (SP) device with three main states of active (A), idle (I) and sleep (S). The definition of the active state is that the SP is in its fully functional state and that it is providing service to some SR. In the idle state, the SP is still fully up and operational, but there are no service requests to deal with, and hence, the SP is in its idle state. The transition between the active and idle states is autonomous, i.e., as soon as the SP completes servicing all of the waiting requests, it enters the idle state. Similarly, the SP goes from idle to active as soon as any service request arrives.

2.2 The Global System Architecture

The architecture of the proposed DPM framework is presented in Figure 1. The frame-work contains one service provider SP which is the component whose power is being managed, i.e. one I/O device. The service requests SR sets for the generating of a typical application class. The Service requests are buffered into a service queue SQ before processing. The exact generating time instances of service requests are considered non stationary.

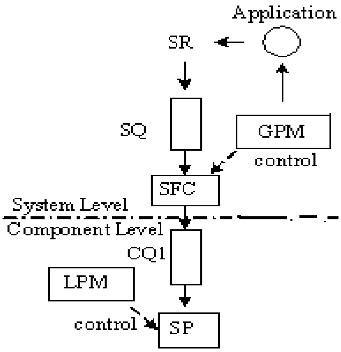


Fig. 1. Block Diagram of a DPM structure

The proposed architecture decomposes the power management task into two layers: component local-level and system global-level. Each service provider is controlled by a local power manager (LPM) whose policy is pre-specified and cannot be changed. We consider Timeout policies for these local self power managed components. The LPM is monitoring the number of service requests that are waiting in the component queue (CQ) and consequently adjusts the state of the service provider.

2.3 The Global Power Manager

At the system-level, the global power manager (GPM) cannot overwrite the LPM policy or directly control the state transition of a service provider. Thus, a Service Flow Control (SFC) is incorporated to control the service request traffic that reaches the SP in order to reduce the global system power consumption while maintaining an acceptable performance. The GPM employs temporal difference learning on semi-Markov decision processes (SMDP) to take power state decision actions.

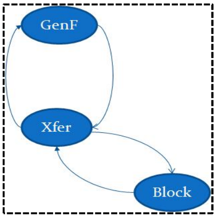


Fig. 2. Model of the SFC

The main functions of the Service Flow Control as shown in figure 2 are detailed as follows:

- **Fake SR Generation:** In case of expecting a lot of activity in the near future, the SFC generates a fake service request (FSR) in order to wake up the SP and prevent it from entering a deep sleep state. An FSR is handled in the same way as a regular SR by the SP, but requires no service from the SP.

- **SR Blocking:** To reduce the wake-up times of the SP and extend the SP sleep time, the SFC blocks all incoming SRs from entering the CQ of the SP. The blocked SRs are stored in the service queue (SQ). Notice that the CQ and SQ are viewed as being identical from the viewpoint of the GPM.
- **SR Transfer:** The SFC continuously moves SRs from the SQ to the CQ, and therefore, the SP will wake up to provide the requested services.

3 Reinforcement Learning For DPM

When an application is running on the CPU, it may send requests to one or more devices for services. These requests are buffered in SQ for processing. The GPM will monitor the SQ and controls the SFC accordingly.

Suppose that we are currently at time t, and the application is running in CPU. Then the state parameters at time t of the power managed component monitored by the GPM are the following three:

1. The SP state, which is the component power state (busy, idle, sleep, etc.).
2. The SQ state, which is the number of requests in the SQ.
3. The SR state, which is the service request generation rate (high, low, medium, etc.) at time t.

3.1 The Local Power Management (LPM) Policy

For the Local Power Management (LPM), the timeout policy is considered as a fixed local policy that cannot be overwritten. For such policy, the SP is put into sleep state if it enters the idle state and remains in that state for more than a specified timeout period. A list of timeout values, as well as immediate shutdown (timeout value = 0), will be considered to show up how the GPM will adapt to the given timeout LPM policies.

3.2 The Global Power Management Policy

The TD(λ) learning algorithm for SMDP presented in [10] is used to learn the optimal Global Power Management (GMP) policy. To apply the RL-based approach for our DPM framework, the decision epochs are first defined. Let N denote the number of waiting requests in the SQ. For this case, the GPM makes decisions in the following four cases:

1. The SP is in the sleep state and SQ contains less than N requests. In this case, the GPM decides to keep the SP in the sleep state and issues ‘SR blocking’ commands to SFC to block all incoming requests from entering the component queue.

2. The SP is in the sleep state and SQ contains N requests. Therefore, the GPM decides to turn on the SP for processing requests. It will issue 'SR transfer' commands to SFC to transfer SRs from the SQ to the CQ, and therefore, the SP will wake up for servicing the requested services.
3. The SP is in the idle state and the timeout has not yet expired:
    a. If some request comes during that period of time. The GPM will issue 'SR transfer' commands to SFC to transfer the incoming SRs from the SQ to the CQ so that the SP goes to the active state for processing re-quests according to the LPM policy.
    b. If it is considered that the timeout will be reached very soon and the GPM is predicting a grouping of N requests to come in the near future after the timeout expires, then the GPM will decide to prevent the SP from entering the sleep state. More precisely, the SFC generates a 'Fake SR' and the SP is kept in the idle state. The estimation of the near future for a grouping of N requests to come relatively after the timeout expiry is controlled based on a user-defined parameter. Let ε denote this parameter.
4. The SP is in the idle state and the timeout has expired, than the SP goes to the sleep state according to its LPM policy.

3.3 The Power-Latency Tradeoff Curve

In this work, we use "cost rate" instead of "reward rate" in RL algorithms [5]. The cost rate is a linearly weighted combination of instantaneous power consumption and the number of requests buffered in the SQ. The relative weight between power and latency can be changed to get a power-latency tradeoff curve.

3.4 Multiple Power-Latency Tradeoff Curves for Varying ε

As stated in the RL based GPM algorithm, when the SP is in the idle state and no request is coming before the timeout expires. If the GPM is predicting a lot of activity in an ε time after the timeout expires (at time t=timeout +ε). Then, According to the proposed policy, 'Fake SR' is generated to prevent the SP from entering the sleep state. Actually, such action results in forcing the SP to wait for requests till timeout +ε which is somewhat equivalent to forcing a new timeout value that corresponds to the sum of the ε value and the original pre-specified timeout fixed by the LPM.

For a fixed ε value, the relative weight between power and latency can be changed to get a certain power-latency tradeoff curve. Now, for varying ε values, multiple power-latency tradeoff curves are obtained. After many trial and fail experiments, the best ε value is obtained for the optimal power latency tradeoff curve.

3.5 Learning ε

The use of multiple ε values enables us to do provide many power-latency tradeoff curves. However, instead of varying manually ε to get the optimal value, the

GPM policy can be enhanced to make it automatically learn the best ε value for a given tradeoff weight between power and latency. In that case, the GPM will learn the best N number of requests to serve and also the best ε value for a fixed power-latency tradeoff.

4 Experimental Results

In this section we present the results of the proposed RL HPM on a wireless adapter card (WLAN card) with real workloads. Table 1 lists the power and delay characteristics of this device. In this table Ttr is the time taken in transitioning to and from the sleep state while Etr is the energy consumed in waking up the device. Tbe refers to the break even time. For the workloads, we have measured several real traces using the tcpdump utility in Linux. including 45-minute trace for online video watching, 2-hours trace for web surfing, and 6-hour trace for a combination of web surfing, online chatting and server accessing, 2 hours each.

Table 1. Power and Delay characteristic of WLAN card

Sleep	*Pbusy*	*Pidle*	*Etr*	*Ttr*	*Tbe*
0.13 W	2.15 W	0.9 W	7.0 J	1.6s	6.8s

To compare the effectiveness of the basic GPM with the fixed Timeout policy we conduct simulation using one type of application with service request trace given by the 6-hour combined trace, similar to the experimental setups in the reference works [10]. The timeout policy of the LPM is fixed to 0.3 Tbe. The action set used correspond to the num-ber of waiting requests in the SQ. For this experiment, the sate-action values considered are listed as follows :{ 1, 2, 3, 4 and 5}.

Table 2. Simulation Results of the basic-GPM on the WLAN-card

	Weight	*HPM* *ε=0.1 Tbe*	*HPM* *ε=0.6 Tbe*	*HPM* *ε=0.9 Tbe*
Power	0.01	1.5348	1.2264	1.1541
Latency	0.99	2.3138	1.3305	1.2110
Power	0.83	1.1556	1.1219	1.0970
Latency	0.17	3.2616	2.2407	1.7522
Power	0.93	0.7308	0.7806	0.7740
Latency	0.07	5.4149	4.9717	4.9918
Power	0.99	0.6244	0.6394	0.6202
Latency	0.01	6.6838	6.2558	6.7105
Power saving % LPM			11.25%	24.25%
Maximum Power saving		71.81%	70.31%	72.23%

Results are presented in Table 2. In this table, three GPM policy with different epsilon's values are compared. The epsilon values are set to be 0.1 Tbe, 0.6 Tbe and 0.9 Tbe. For each GPM policy with a fixed epsilon, the relative weight between power and latency can be adjusted to get the desired power-latency tradeoff. We performed different simulations using different weights and we reported the average power consumption of the SP and the average latency per request, the power saving in comparison to the fixed Timeout (using the latency obtained with the fixed timeout policy). The maximum power saving for each GPM obtained for varying weights is given in the last row. It can be seen that the basic GPM provides a wide range of power-latency tradeoffs. Even with the same latency, the proposed RL-based GPM achieves lower power consumption than the fixed Timeout. It even outperforms reference [16] by 24.25% in terms of energy consumption saving. The maximum of energy improvement of the LPM controlled service provider is obtained for low delay constraints system and is about 72% of energy saving.

To better understand how the energy savings are distributed for different power-latency tradeoffs, we reported in figure 3 the power and latency tradeoff curves that are obtained using the pre-specified timeout policy, and the basic GPM with two different ε values fixed to 0.1 Tbe and 0.6 Tbe. Using these curves, we can see that the proposed RL-basic GPM has the same behavior for high delay values. However, with ε value equal to 0.6 Tbe the PM can further minimize power in comparison to what is achieved with ε value equal to 0.1 Tbe.

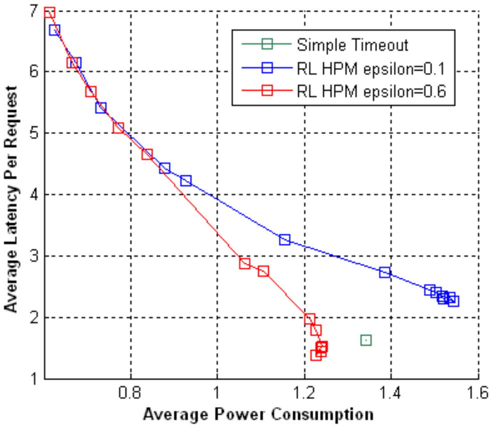


Fig. 3. Power- Latency tradeoff curves of the Fixed Timeout and the basic RL GPM on WLAN card with ε=0.1 and ε=0.6

The second set of experiment shows the effectiveness of the RL-Learning epsilon HPM in comparison to what it is obtained using the basic framework. We conduct simulation using the same WLAN card and the same application type. Figure 4 presents the power-latency tradeoff curves for both ε=0.1 Tbe, ε=0.6 Tbe, values and for the learned epsilon. It can be seen RL-Learned epsilon GPM can achieve much lower power consumption than the RL-Basic DPM particularly for high constrained delay systems.

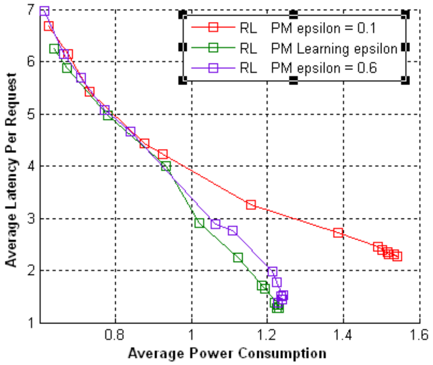


Fig. 4. The effectiveness of learning epsilon on WLAN card

5 Conclusions

This paper presents a Dynamic Power Management framework based on reinforcement learning (RL) technique which aims to save power in an Energy-Managed Computer (EMC) system with self power-managed components. The proposed approach consists of two layers: component-level and system–level global power manager. The component-level PM policy is a timeout pre-specified policy whereas the system-level global PM perform power management in a continuous-time and event-driven manner using temporal difference learning on Semi-Markov Decision Process (SMDP) for model-free Reinforcement Learning (RL) techniques. The TD(λ) for SMDP problems is used as the basic RL algorithm in the proposed system. The proposed HPM is model-free and requires no prior information of the workload characteristics. Experiments show that that the proposed HPM scheme enhances power savings considerably while maintaining a good performance level. In comparison with other reference systems, the proposed RL DPM approach performs well under various workloads, can simultaneously consider power and performance and achieves a wide and deep power-performance tradeoff curves.

References

1. Benini, L., Bogliolo, A., De Micheli, G.: A survey of design techniques for system level dynamic power management. IEEE Trans. on VLSI Systems 8(3), 299–316 (2000)
2. Srivastava, M., Chandrakasan, A., Brodersen, R.: Predictive system shutdown and other architectural techniques for energy efficient programmable computation. IEEE Trans. on VLSI (1996)
3. Hwang, C.H., Wu, A.C.: A predictive system shutdown method for energy saving of event-driven computation. In: ICCAD 1997 (1997)
4. Benini, L., Paleologo, G., Bogliolo, A., De Micheli, G.: Policy optimization for dynamic power management. IEEE Trans. on CAD 18, 813–833 (1999)

5. Qiu, Q., Pedram, M.: Dynamic Power Management Based on Continuous-Time Markov Decision Processes. In: DAC 1999 (1999)
6. Simunic, T., Benini, L., Glynn, P., De Micheli, G.: Event-driven power management. IEEE Trans. on CAD (2001)
7. Jung, H., Pedram, M.: Dynamic power management under uncertain information. In: DATE 2007, pp. 1060–1065 (April 2007)
8. Dhiman, G., Simunic Rosing, T.: Dynamic power management using machine learning. In: ICCAD 2006, pp. 747–754 (November 2006)
9. Tan, Y., Liu, W., Qiu, Q.: Adaptive Power Management Using Reinforcement Learning. In: ICCAD 2009, pp. 461–467 (November 2009)
10. Wang, Y., Xie, Q., Ammari, A.C., Pedram, M.: Deriving a near-optimal power management policy using model-free reinforcement learning and Bayesian classification. In: DAC 2011, pp. 875–878 (June 2011)
11. Bradtke, S., Duff, M.: Reinforcement learning methods for continuous-time Markov decision problems. In: Advances in Neural Information Processing Systems 7, pp. 393–400. MIT Press (1995)
12. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (August 2006)
13. Lu, Y.-H., Benini, L., De Micheli, G.: Power-aware operating systems for interactive systems. IEEE Trans. VLSI System 10(4), 119–134 (2002)
14. Simunic, T., Boyd, S.: Managing power consumption in networks on chips. In: DATE 2002 (2002)
15. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
16. Rong, P., Pedram, M.: A Stochastic Framework for Hierarchical System-Level Power Management. In: Proc. of Symp. on Low Power Electronics and Design, pp. 269–274 (August 2005)
17. Ren, Z., Krogh, B.H., Marculescu, R.: Hierarchical adaptive dynamic power management. IEEE Trans. Computers 54(4), 409–420 (2005)

Network Time Synchronization: A Full Hardware Approach

Jorge Juan, Julian Viejo, and Manuel J. Bellido

Departamento de Tecnología Electrónica,
ETSI Informática, Universidad de Sevilla, Spain
jjchico@dte.us.es
http://www.dte.us.es/id2

Abstract. Complex digital systems are typically built on top of several abstraction levels: digital, RTL, computer, operating system and software application. Each abstraction level greatly facilitates the design task at the cost of paying in performance and hardware resources usage. Network time synchronization is a good example of a complex system using several abstraction levels since the traditional solutions are a software application running on top of several software and hardware layers. In this contribution we study the case where a standards-compliant network time synchronization solution is fully implemented in hardware on a FPGA chip doing without any software layer. This solution makes it possible to implement very compact, inexpensive and accurate synchronization systems to be used either stand-alone or as embedded cores. Some general aspects of the design experience are commented together with some figures of merit. As a conclusion, full hardware implementations of complex digital systems should be seen as a feasible design option, from which great performance advantages can be expected, provided that we can find a suitable set of tools and control the design development costs.

Keywords: digital systems, hardware, network time synchronization, FPGA.

1 Introduction

Complex digital systems are typically built on top of several abstraction levels: digital, RTL, computer, operating system and software application. Each abstraction level, together with design automation tools, greatly facilitates the design task at the cost of the overhead introduced by every abstraction layer. This is payed in the form of reduced performance (both timing and power) and a much higher hardware resources usage. However, some critical parts in complex digital systems still require a low level implementation in order to improve performance or reduce power consumption. This is the case of the numerous hardware accelerators used today for audio and video processing that can be found in high performance or resource-limited devices like graphic adapters or

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 225–234, 2013.

smart phones. At the same time, there are an increasing number of devices implementing in hardware high level functions typically done in software, like the QuickTCP IP core from PLDA [1]. However, there is still plenty of room for performance improvement in some high-level traditional software applications that can get a big performance boost if fully implemented in hardware. These applications can be improved by several orders of magnitude in speed, lower power consumption and smaller hardware footprint (area, logic blocks, etc.). These gains are a consequence of doing without some of the abstraction layers like the software and computer abstractions.

A full hardware implementation has to face a number of challenges like handling the complexity at the low level, design tool availability, development cost and time-to-market. Design teams also need to believe that it can be done. Network time synchronization is a very good example of this kind of traditional software applications where we can find clients, servers, network protocols and elaborated processing algorithms. This contribution summarizes the implementation of an prototype embedded network time synchronization system that is being developed completely in hardware. Some preliminary results show that the system can be fully implemented in hardware and perform with an accuracy comparable to commercial industrial equipment at a fraction of the cost and resources usage. The system can be implemented stand-alone or as a soft-core in a low grade FPGA chip.

In the following section, a brief introduction to network time synchronization is given. Section 3 is an overview of the synchronization system while Sect. 4 describes the design platform and tools. Section 5 summarizes some figures of merit and some conclusions are derived in Sect. 6.

2 Network Synchronization

The two main network synchronization protocols in use today are the Network Time Protocol (NTP) [2], and the Precision Time Protocol (PTP) [3]. NTP is more used to synchronize Internet equipment and is used by almost any Internet router and server. NTP servers typically serve thousands of clients over world wide network connections so the poll intervals are in the range of several minutes to a few hours and the network latencies are difficult to predict. To cope with that, NTP includes sophisticated mitigation and clock disciplining algorithms achieving a clock accuracy in the range of the millisecond. On the other hand, PTP was designed to synchronize equipment in industrial environments connected to a local area network and aims to sub-microsecond clock accuracy, making it suitable for measurement and control systems. PTP uses poll intervals in the range of a second or lower and most implementations uses some kind of dedicated hardware to implement the most critical parts and gain accuracy. The NTP standard also defines the Simple Network Time Protocol (SNTP) a simplified version of NTP, that do not impose the use of the mitigation and clock disciplining algorithms found in NTP. SNTP servers are typically primary servers connected to a single reference clock and SNTP clients typically connect to a single servers and do not have dependent clients and have been used in scenarios similar to PTP. Otherwise, NTP and SNTP clients and servers can inter-operate seamlessly.

Both NTP and PTP synchronization is based in the interchange of packets between clients and servers. This mechanism is called the on-wire protocol, which objective is to determine the offset of the local clock at the client with respect to the server and the latency of the network connection. Both NTP and PTP on-wire protocols are very similar. Here we describe the operation of the NTP/SNTP on-wire protocol (Fig. 1). The client sends a request to the server by issuing an UDP data packet where the time of its local clock (T_1) is included. When the request is received at the server a new time stamp T_2 is generated with the reception time as given by the server's local clock. After processing the request, the server issues a reply including the time at which the reply leaves the server (T_3). When the client receives the reply the arrival time (T_4) is also annotated. With this set of timestamps the client can calculate the round trip time (t_{rd}) and the time offset between the server's and client's clocks (t_{offset}). Assuming a symmetric connection these times can be calculated according to eq. (1).

$$\begin{aligned} t_{rd} &= (T_4 - T_1) - (T_3 - T_2) \\ t_{offset} &= \frac{(T_2 - T_1) + (T_3 - T_4)}{2}. \end{aligned} \tag{1}$$

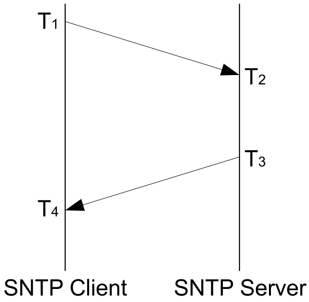


Fig. 1. On-wire (S)NTP protocol operation

Using the calculated offset the client can correct its local clock to match the server time. Software implementations of NTP tipically achieve time synchronization within a millisecond with respect to the server [2,4]. There are two main sources of error. The first one is the asymmetry in the network communication when the time spent by the client's request to reach the server is different to the time spent by the answer to reach the client. This is due to unpredictable latency in network equipment, specially when collisions take place and the number of the devices involved increases. The second main source of error is due to the variable time gap between the instant the time stamp is registered in the datagram and the real instant the datagram leaves or reaches the host. In typical software implementation, these time stamps are registered by client/server software running as a user level application so the time stamp error will depend on the time spent processing the datagram as it goes through the protocol stack and software layers. This error will largely depend on system load, detailed software implementation, etc.

As in PTP, the precision of the NTP synchronization can be largely improved by doing the time-stamping operation in lower layers [5,6,7]. The highest precision in the time-stamping operation is only achievable if done by the Ethernet device hardware as soon as the packets arrive or leave the interface. Thus, precision at the time client can be better than one microsecond [7].

3 Synchronization System Overview

The objective of the project leading to the design experience summarized in this contribution was to design a very compact and low cost SNTP client and server suitable for the scenario depicted in Fig. 2, where the SNTP server gathers the time from the GPS receiver and SNTP clients provide time and synchronization information to remote terminal units (RTU's). The SNTP server will use a standard GPS receiver as a time reference. It will use the PPS (Pulse Per Second) signal and NMEA [8] data from the GPS to synchronize its internal clock. The SNTP clients will synchronize with the server through the local area network using the NTP protocol, and will provide a PPS signal and NMEA information through a serial interface, emulating a GPS receiver. With this solution, there is no need to provide a GPS receiver with every RTU to avoid extra costs, complexity and solve the cases where the installation of a GPS antenna is not feasible. A summary of the system specifications follows:

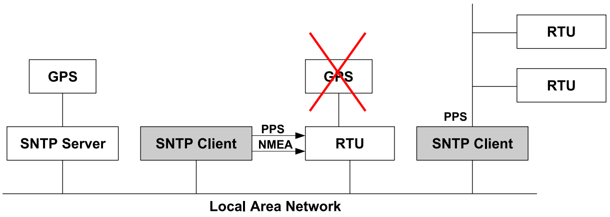


Fig. 2. Synchronization system overview

- Clients and servers should operate in a standard 10/100/1000 MHz Ethernet LAN.
- Clients and servers should gather their configuration parameters automatically using the BOOTP protocol [9] so that the configuration for all the clients and servers can be centralized in a single BOOTP server.
- The precision of the local clock at the clients and the server should be within 10 μs of a GPS reference in optimal conditions: low network load and direct LAN connection without switches. In typical conditions precision should be always within 1 ms.

- The whole client and server designs should fit in a single, low density FPGA chip and should need no additional hardware, so that the system can work stand-alone or embedded in a bigger system.
- Low power: implemented in a low density, low frequency FPGA, the client or server should consume under 1 W of average power.
- System clock frequency is 50 MHz.

After thoughtful consideration, it was decided to implement both the client and server completely in hardware without a software abstraction level. The most important reasons leading to this decision were that in order to achieve a high accuracy, some of the key parts of the system, like timestamping, need be done in hardware anyway; and that the lack of a processor and associated subsystems (RAM, file system, etc.) should give a good footprint in terms of resources and power consumption. In the rest of the section, the most important aspects of the design and implementation are commented. Diagrams of the modules that form the SNTP server and client are shown in Fig. 3. Server and client share the Ethernet MAC controller and the UART with no modifications. The protocol and configuration interface and the synchronization module are the most complex blocks and most of its functionality is shared between the client and the server. The transmission and receptions modules are specific to the server and client respectively. Next, we will briefly describe the functionality of these block.

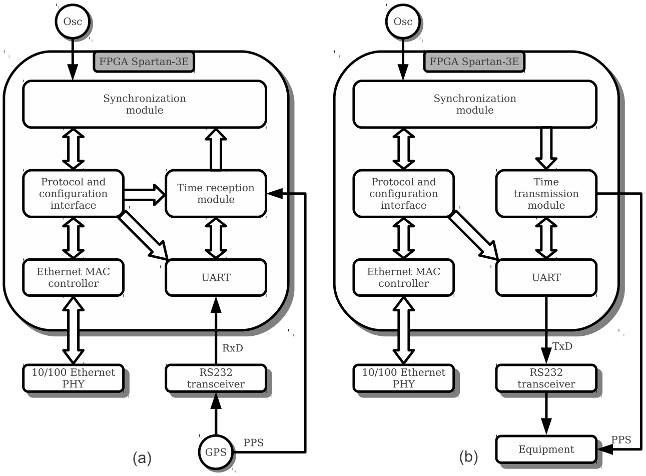


Fig. 3. SNTP client and server block diagram. a) server, b) client

The protocol and configuration interface is probably the most complex block in the system. It is in charge of handling the configuration process and the various communication protocols involved during the configuration and normal operation phases: BOOTP, IP, ARP, UDP and NTP. It also acts as the control unit of the system since many system tasks are triggered by the protocol interface. At startup time, the configuration phase starts by requesting a configuration packet through the BOOTP protocol. A BOOTP response provides the client with an IP network address and mask client configuration parameters like the NTP server address, the poll interval, UART baud rate and some internal clock tunning parameters. After configuration, the interface enters normal operation and issues NTP requests at the configured poll intervals, and collect the responses. It is also in charge of registering and reading the timestamps of the NTP packets as they are processed. This information is then transferred to the synchronization module to make the necessary clock adjustments.

The task of the synchronization module is to maintain the local clock time as accurate as possible. In the client, when a NTP response arrives, the timestamps are transferred to the synchronization module that calculates the local clock offset according to eq. (1). In the server, the offset is calculated by using the reference provided by the GPS trhough the reception module. Then, the local time is corrected by introducing slight frequency variations to the local clock counter, so that a good frequency stability is achieved. The frequency control is done with the clock discipline algorithm published in [11] which provides both a high accurate control and a low time to synchronization. By design, the local clock resolution for the fractional part of the second is 22 bits (238 ns time resolution) which is intended to provide local clock accuracy in the range of 1 μs.

The transmission (server) and reception (client) modules handles the conversion between the local clock time format (that follows NTP standards) and NMEA-0183 Recommended Minimum sentence C (RMC) [8] frame format used to communicate with the GPS unit in the server, and with the external equipment in the client. The Ethernet MAC controller provides the standard functionality and is in charge of controlling a standard Fast Ethernet PHY device, allowing the transmission and reception of Ethernet frames conforming to IEEE 802.3 specification.

4 Platform and Tools

The systems are implemented on FPGA chips from Xilinx using Xilinx tools. Development has been done in a Digilent's Spartan 3E Starter Board [12] that includes a mid-range Xilinx Spartan-3E XC3S500E FPGA chip. Most of the design is coded in hardware description languages (mostly Verilog but also some VHDL) that has been synthesized using Xilinx's XST synthesizer. Xilinx's System Generator for DSP (SGDSP) [13] has also been extensively used to implement complex arithmetic operations.

The Ethernet MAC controller is the Tri-mode Ethernet MAC IP-core available from the OpenCores web portal [14]. The block is available as Verilog code and has been slightly customized to make a more efficient memory usage of the FPGA resources. The top-level design in the protocol and configuration interface and the synchronization module have been designed using SGDSP. This has facilitated a fast design and interconnection of processing blocks, registers and memories. Controlling units in these blocks have been designed as state machines coded in Verilog as black boxes inside the SGDSP project. The reception and transmission modules have been implemented using custom time format converters coded in VHDL and a simple processing unit implemented using Picoblaze [15], a very simple soft microprocessor from Xilinx.

Verilog and VHDL code has been simulated using Xilinx's ISIM logic simulator included with the ISE design suit [16]. Inside SGDSP, ISIM is also used to simulate the user's black boxes. Otherwise, the high level simulation capabilities of SGDSP, that uses Matlab/Simulink have been extensively exploited for the associated modules, using various data sources and sinks (scopes) provided by SGDSP and custom Matlab scripts, which has greatly facilitated the simulation process. Xilinx's ChipScope [17] integrated logic analyzer has also been used for testing the FPGA chip during the first part of the development, specially to test and validate the clock discipline algorithms. Later, a custom logic analyzer named LEA (Logic Event Analyzer) [18] was developed for the long-term testing of the system to overcome the storage capabilities of ChipScope.

5 Results

In this section some preliminary results are presented in order to evaluate the system performance and fulfillment of specifications. First, Table 1 summarizes the hardware resources consumed by the implementation on a Xilinx's XC3S500E chip. Both client and server designs occupies less than 50% of the chip's resources except for the number of slices. Considering this is a low grade family we can say that hardware footprint is reasonably low and would be under the 10% of slice occupation in any mid-range Virtex-6 family of FPGA's of the same vendor. The resources are slightly higher in the client because of the transmission module, which needs a divider and other arithmetic blocks with an extra complexity when compared to the server's reception module.

Table 1. Hardware implementation results (FPGA Spartan-3E XC3S500E)

Resource	SNTP Client - Use (%)	SNTP Server - Use (%)
Slices	3615 (77 %)	2927 (62 %)
Slice Flip Flops	3753 (40 %)	2941 (31 %)
4-input LUT	4475 (48 %)	3424 (36 %)
RAMB16	5 (25 %)	5 (25 %)

Table 2 shows the accuracy of three servers as seen by the standard NTP software client running in a personal computer. The Internet server is a public NTP server available and located in another country, the commercial server Lantime M600 Network Time Server from Meinberg [19] and the prototype is an implementation of the full hardware SNTP server described in this contribution.

Both the commercial server and the prototype are connected to the same LAN than the software client. Delay measures the round-trip-time from the client to the server, offset is the displacement of the local clock and jitter measures the stability of the synchronization. As expected, both local servers provide much better synchronization than the Internet server. Also, both local servers give similar quality figures as seen by the client with slightly better results for our prototype.

Table 2. Software client synchronization results: estimated delay, offset and jitter, all in milliseconds

	Delay	Offset	Jitter
Internet server	33.89	-2.410	0.550
Commercial server	0.251	-0.044	0.054
Prototype	0.101	-0.013	0.045

Table 3 shows the mean offset and offset error as seen by the hardware client when synchronized to the hardware server operating in the same LAN, for various combinations of the poll interval exponent (p) and the attenuation factor (q). p controls the poll interval which is 2^p and q controls how aggressively the offset is corrected with softer corrections for higher values of q [11]. Nominal values of the implementation are $p = 0$ and $q = 2$. This mainly shows that the discipline algorithms implemented in the client are able to maintain a local clock accuracy below 1 μs which surpasses the initial specification of 10 μs.

Table 3. Hardware client-server synchronization. Mean offset ± error in microseconds.

	$q = 0$	$q = 1$	$q = 2$	$q = 3$
$p = 0$	0.06 ± 1.21	0.06 ± 1.17	0.07 ± 0.95	0.05 ± 0.85
$p = 2$	0.07 ± 1.71	0.05 ± 1.51	0.07 ± 1.69	0.20 ± 1.81
$p = 4$	0.04 ± 3.88	0.23 ± 4.34	1.01 ± 6.53	0.55 ± 14.86
$p = 6$	0.24 ± 13.18	0.59 ± 17.97	1.16 ± 35.29	9.19 ± 64.45

In order to test the performance of the server prototype it is loaded with a varying number of requests per second (rps) by injecting NTP traffic in the LAN. At the same time, the mean offset and offset error is collected from a client prototype. It has been checked that the synchronization accuracy is not affected with a low number of rps and that the server can easily handle 10000 rps maintaining an accuracy of 3 μs with a maximum rps estimated above 40000 rps. Typically, Meinberg equipment specifies a maximum of 10000 rps without mentioning the expected accuracy, while Symmetricom's [20] announces a time stamp accuracy of 14 μs under 3200 rps.

Finally, Table 4 estimates the power consumption and unit cost of various NTP server alternatives: a commercial server like the ones from Meinberg or Symmetricom, a software server implemented in an embedded computer like the Beagleboard [21] and our hardware SNTP server prototype. The stand-alone prototype power consumption is measured on a custom printed circuit board (PCB) that includes the FPGA chip and all the necessary peripherals. The reduced power consumption of the prototype is due to its much more simple hardware architecture compared to the other alternatives. The unit cost is also reduced since the prototype does not need additional devices like flash or RAM memory. Actually, if the prototype design is embedded in a bigger system as a soft IP-core, the power consumption and cost can be almost negligible.

Table 4. Power consumption and unit cost estimations for various NTP server implementation alternatives

	Commercial server	Embedded comp.	Prototype (stand-alone)	Prototype (embedded)
P (W)	20.4	$3,11$	0.624	≈ 0
Unit cost ($)	$6000 - 8000$	180	50	≈ 0

6 Conclusions

This contribution summarizes the implementation of a an embedded network time synchronization system completely implemented in FPGA hardware to illustrate the feasibility a full hardware implementation of high level functions typically found in the software layer. The accuracy of the system is in the same range of commercial equipment while the needed resources, power and cost is two orders of magnitude lower, at the cost of sacrificing some additional functionality and flexibility. The design is carried out using the Xilinx's tool set including DSP libraries, but only standard blocks and functions are used so it could be completely ported to a hardware description language to achieve vendor and technology independence.

This is a good study-case to supports the idea that high-level system functions tightly related to software like network protocols and network synchronization can be completely ported to hardware to obtain a very cheap yet much higher performance solution. This gain is mainly due to the simplification of the problem by removing some abstraction layers like the computer and the software abstractions. Newer design tools and the conception of a hardware operating system-like abstraction layer can make the full hardware approach widely available to a range of traditional software applications.

Acknowledgments. This work has been partially supported by the Ministerio de Ciencia e Innovacin of the Spanish Government under project TEC2011-27936 (HIPERSYS) and by the European Regional Development Found (ERDF).

References

1. PLDA: PLDA Home Page
2. Mills, D.L., Martin, J., Burbank, J., Kasch, W.: Network Time Protocol Version 4: Protocol and Algorithms Specification. RFC 5905 (Standards Track) (June 2010)
3. IEEE: IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. PTP Version 2 (1588-2008) (2008)
4. Mills, D.L.: Computer Network Time Synchronization: The Network Time Protocol. CRC Press, Inc., Boca Raton (2006)
5. Skeie, T., Johannessen, S., Løkstad, T., Holmeide, Ø.: Same time - Different place. ABB Review (2), 9–14 (2003)
6. Johannessen, S.: Time Synchronization in a Local Area Network. IEEE Control Systems Magazine 24(2), 61–69 (2004)
7. Holmeide, Ø., Skeie, T.: Synchronised: Switching. IET Computing and Control Engineering 17(2), 42–47 (2006)
8. NMEA: NMEA 0183 Standard. NMEA 0183 V 4.00 (January 2002)
9. Croft, W.J., Gilmore, J.: Bootstrap Protocol. RFC 951 (Draft Standard) (September 1985) Updated by RFCs 1395, 1497, 1532, 1542
10. Plummer, D.: Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware. RFC 826 (Standard) (November 1982) Updated by RFC 5227
11. Viejo, J., Juan, J., Bellido, M., Millan, A., Ruiz-de Clavijo, P.: Fast-convergence microsecond-accurate clock discipline algorithm for hardware implementation. IEEE Transactions on Instrumentation and Measurement 60(12), 3961–3963 (2011)
12. Digilent: Digilent Home Page
13. Xilinx: System Generator for DSP Getting Started Guide Release 10.1. Xilinx, Inc. (March 2008)
14. OpenCores: OpenCores Home Page
15. Chapman, K.: PicoBlaze 8-Bit Embedded Microcontroller User Guide for Spartan-3, Virtex-II and Virtex-II PRO FPGAs. Xilinx, Inc. (November 2005)
16. Xilinx: ISE In-Depth Tutorial. Xilinx, Inc. (March 2011)
17. Xilinx: ChipScope Pro 11.1 Software and Cores User Guide. Xilinx, Inc. (April 2009)
18. Viejo, J., Villar, J., Juan, J., Millan, A., Ostua, E., Quiros, J.: Long-term on-chip verification of systems with logical events scattered in time. Microprocessors and Microsystems 33(5), 402–408 (2012)
19. Meinberg: Meinberg Funkuhren GmbH & Co. KG Home Page
20. Symmetricom: Symmetricom Home Page
21. BeagleBoard.org: BeagleBoard Home Page

Case Studies of Logical Computation on Stochastic Bit Streams

Peng Li[1], Weikang Qian[2], David J. Lilja[1], Kia Bazargan[1], and Marc D. Riedel[1]

[1] Electrical and Computer Engineering, University of Minnesota, MN, USA, 55455
{lipeng,lilja,kia,mriedel}@umn.edu
[2] University of Michigan-Shanghai Jiao Tong University Joint Institute, China, 200241
qianwk@sjtu.edu.cn

Abstract. Most digital systems operate on a positional representation of data, such as binary radix. An alternative is to operate on random bit streams where the signal value is encoded by the probability of obtaining a one versus a zero. This representation is much less compact than binary radix. However, complex operations can be performed with very simple logic. Furthermore, since the representation is uniform, with all bits weighted equally, it is highly tolerant of soft errors (i.e., bit flips). Both combinational and sequential constructs have been proposed for operating on stochastic bit streams. Prior work has shown that combinational logic can implement multiplication and scaled addition effectively; linear finite-state machines (FSMs) can implement complex functions such as exponentiation and tanh effectively. Building on these prior results, this paper presents case studies of useful circuit constructs implement with the paradigm of logical computation on stochastic bit streams. Specifically, it describes finite state machine implementations of functions such as edge detection and median filter-based noise reduction.

1 Introduction

In a paradigm advocated by Gaines [1], logical computation is performed on stochastic bit streams: each real-valued number x $(0 \leq x \leq 1)$ is represented by a sequence of random bits, each of which has probability x of being one and probability $1-x$ of being zero. Compared to a binary radix representation, a stochastic representation is not very compact. However, it leads to remarkably simple hardware for complex functions; also it provides very high tolerance to soft errors.

There are two possible coding formats: a unipolar format and a bipolar format [1]. These two coding formats are the same in essence, and can coexist in a single system. In the unipolar coding format, a real number x in the unit interval (i.e., $0 \leq x \leq 1$) corresponds to a bit stream $X(t)$ of length L, where $t = 1, 2, ..., L$. The probability that each bit in the stream is one is $P(X = 1) = x$. For example, the value $x = 0.3$ could be represented by a random stream of bits such as 0100010100, where 30% of the bits are "1" and the remainder are "0." In the bipolar coding format, the range of a real number x is extended to $-1 \leq x \leq 1$. However, the probability that each bit in the stream is one is $P(X = 1) = \frac{x+1}{2}$. The trade-off between these two coding formats is that the bipolar format can deal with negative numbers directly while, given the same bit stream length,

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 235–244, 2013.

L, the precision of the unipolar format is twice that of the bipolar format. (Unless stated otherwise, our examples will use the unipolar format.)

The synthesis strategy is to cast logical computations as arithmetic operations in the probabilistic domain and implement these directly as stochastic operations on datapaths. Two simple arithmetic operations – multiplication and scaled addition – are illustrated in Figure 1.

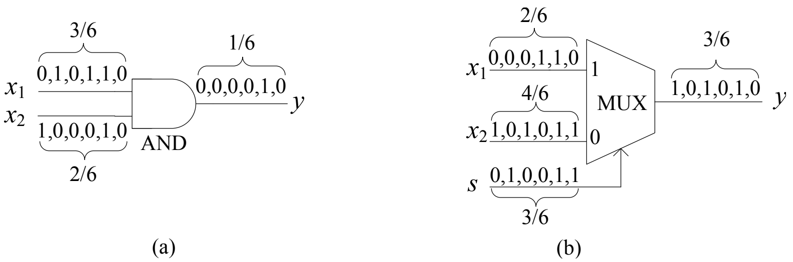


Fig. 1. Stochastic implementation of arithmetic operations: (a) Multiplication; (b) Scaled addition

- **Multiplication.** Consider a two-input AND gate, shown in Figure 1(a). Suppose that its inputs are two independent bit streams X_1 and X_2. Its output is a bit stream Y, where

$$\begin{aligned} y &= P(Y=1) = P(X_1 = 1 \text{ and } X_2 = 1) \\ &= P(X_1 = 1)P(X_2 = 1) = x_1 x_2. \end{aligned}$$

  Thus, the AND gate computes the product of the two input probability values.

- **Scaled Addition.** Consider a two-input multiplexer, shown in Figure 1(b). Suppose that its inputs are two independent stochastic bit streams X_1 and X_2 and its selecting input is a stochastic bit stream S. Its output is a bit stream Y, where

$$\begin{aligned} y &= P(Y=1) \\ &= P(S=1)P(X_1=1) + P(S=0)P(X_2=1) \\ &= s x_1 + (1-s) x_2. \end{aligned}$$

  (Note that throughout the paper, multiplication and addition represent *arithmetic* operations, not Boolean AND and OR.) Thus, the multiplexer computes the scaled addition of the two input probability values.

In the decades since Gaines' original work, there have been numerous papers discussing the paradigm. Most notable has been the work by Brown and Card [2]. They demonstrated efficient constructs for a wide variety of basic functions, including multiplication, squaring, addition, subtraction, and division. Further, they provided elegant constructs for complex functions such as tanh, linear gain, and exponentiation.[1] They

[1] Such functions were of interest to the artificial neural networks community. The tanh function, in particular, performs a non-linear, sigmoidal mapping; this is used to model activation function of a neuron.

used combinational logic to implement simple functions such as multiplication and scaled addition; the used sequential logic in the form of linear finite-state machines (FSMs) to implement complex functions such as tanh.

More recently, Qian et al. presented a general synthesis method for logical computation on stochastic bit streams [3][4][5]. They showed that combinational logic can be synthesized to implement arbitrary polynomial functions, provided that such polynomials map the unit interval onto the unit interval. Their method is based on novel mathematics for manipulating polynomials in a form called Bernstein polynomials. In [4] Qian et al. showed how to convert a general power-form polynomial into a Bernstein polynomial with coefficients in the unit interval. In [3] they showed how to realize such a polynomial with a form of "generalized multiplexing." In [5], they demonstrated a reconfigurable architecture for computation on stochastic bit streams. They analyzed cost as well as the sources of error: approximation, quantization, and random fluctuations; also they studied the effectiveness of the architecture on a collection of benchmarks for image processing. Li and Lilja demonstrated a stochastic implementation of a kernel density estimation-based image segmentation algorithm [6].

After an introduction to the concepts and a review of implementations of functions such as tanh and exponentiation, this paper presents case studies of useful circuit constructs implemented with the paradigm of logical computation on stochastic bit streams. Specifically, it describe finite state machine implementations of functions for image processing such as edge detection and median filter-based noise reduction.

1.1 Stochastic Exponentiation Function

When operating on stochastic bit streams, combinational logic can only implement polynomial functions of a specific form – namely those that map the unit interval to the unit interval [4]. Non-polynomial functions can be approximated by combinational logic, for instance with MacLaurin expansions [3]. However, highly non-linear functions such as exponentiation and tanh cannot be approximated effectively with this approach. This limitation stems from the fact combinational logic can only implement scaled addition in the stochastic paradigm. The implementation of polynomials with coefficients not in the unit interval is sometimes not possible and is generally not straightforward [5].

Gaines [1] described the use of an ADaptive DIgital Element (ADDIE) for generation of arbitrary functions. The ADDIE is based on a saturating counter, that is, a counter which will not increment beyond its maximum state value or decrement below its minimum state value. In the ADDIE, the state of the counter is controlled in a closed loop fashion. The problem is that ADDIE requires that the output of the counter to be converted into a stochastic bit stream in order to implement the closed loop feedback [1]. This is potentially inefficient and hardware intensive.

In 2001, Brown and Card [2] presented the stochastic exponentiation (SExp) function, with the state transition diagram shown in Figure 2. This configuration approximates an exponentiation function stochastically as follows,

$$y \approx \begin{cases} e^{-2Gx}, & 0 \leq x \leq 1, \\ 1, & -1 \leq x < 0, \end{cases} \tag{1}$$

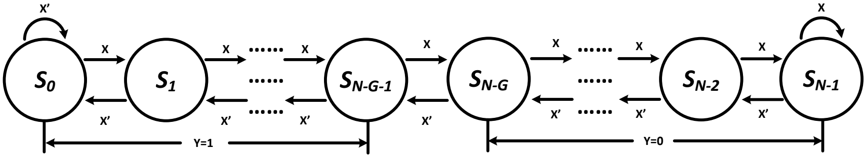


Fig. 2. State transition diagram of the FSM-based stochastic exponentiation function

where x is the bipolar encoding of the input bit stream X and y is the unipolar encoding of the output bit stream Y.

The FSM shown in Figure 2 is similar to Gaines' ADDIE. The difference is that this linear FSM does not use a closed loop [1][2]; accordingly this construct is much more efficient.

1.2 Scaled Subtraction

The scaled subtraction can be implemented with a MUX and a NOT gate, as shown in Fig. 3.

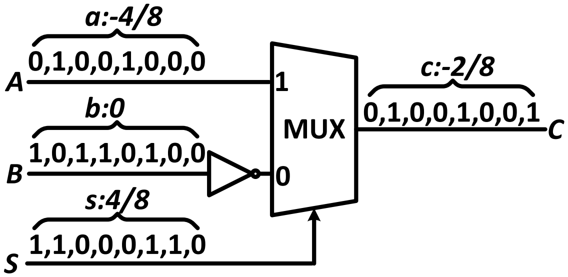


Fig. 3. Scaled subtraction with the bipolar coding. Here the inputs are $a = -4/8$ and $b = 0$. The scaling factor is $s = 4/8$. The output is $4/8 \times (-4/8) + (1 - 4/8) \times 0 = -2/8$, as expected.

The scaled subtraction only works for bipolar coding, since subtraction can result negative output value and the unipolar coding format cannot represent negative values. Similar to the case of scaled addition with the bipolar coding, the stochastic bit streams A, B, and C use the bipolar coding format and the stochastic bit stream S uses the unipolar coding format, i.e.,

$$\begin{aligned} a &= 2P(A = 1) - 1, \\ b &= 2P(B = 1) - 1, \\ s &= P(S = 1), \\ c &= 2P(C = 1) - 1. \end{aligned}$$

Based on the logic function of the circuit, we have

$$\begin{aligned} P(C=1) &= P(S=1)\cdot P(A=1) \\ &\quad + P(S=0)\cdot P(B=0), \end{aligned}$$

i.e.,

$$\frac{c+1}{2} = s\cdot\frac{a+1}{2} + (1-s)\cdot\frac{1-b}{2}.$$

Thus, we have $c = s\cdot a - (1-s)\cdot b$. It can be seen that, with the bipolar coding format, the computation performed by a MUX and a NOT gate is the scaled subtraction of the two input values a and b, with a scaling factor of s for a and $1-s$ for b.

1.3 Stochastic Tanh Function

The stochastic tanh function is also developed by Brown and Card [2]. The state transition diagram of the FSM implementing this function is shown in Fig. 4. If x and y are the bipolar coding of the bit streams X and Y, respectively, i.e., $x = 2P_X - 1$ and $y = 2P_Y - 1$, Brown and Card proposed that the relationship between x and y was,

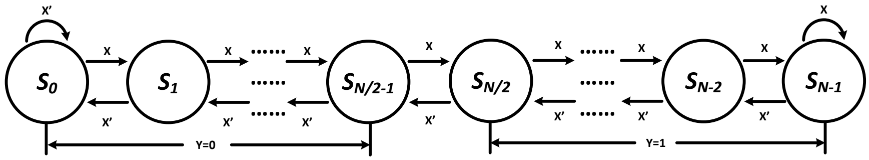


Fig. 4. State transition diagram of the FSM implementing the stochastic tanh function

$$y = \frac{e^{\frac{N}{2}x} - e^{-\frac{N}{2}x}}{e^{\frac{N}{2}x} + e^{-\frac{N}{2}x}}. \tag{2}$$

The corresponding proof can be found in [7]. In addition, Li and Lilja [7] proposed to use this function to implement a stochastic comparator. Indeed, the stochastic tanh function approximates a threshold function as follows if N approaches infinity,

$$\lim_{N\to\infty} P_Y = \begin{cases} 0, & 0 \le P_X < 0.5, \\ 0.5, & P_X = 0.5, \\ 1, & 0.5 < P_X \le 1. \end{cases}$$

The stochastic comparator is built based on the stochastic tanh function and the scaled subtraction as shown in Fig. 5. $P_S = 0.5$ in the selection bit S of the MUX stands for a stochastic bit stream in which half of its bits are ones. Note that the input of the stochastic tanh function is the output of the scaled subtraction. Based on this relationship, the function of the circuit shown in Fig. 5 is:

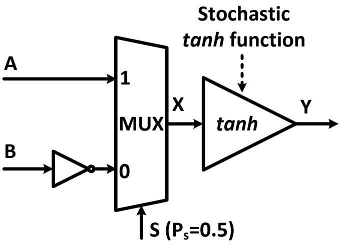


Fig. 5. The stochastic comparator

$$\boxed{\textit{if}\,(P_A < P_B)\ \textit{then}\ P_Y \approx 0;\ \textit{else}\ P_Y \approx 1,}$$

where P_A, P_B, and P_Y are the probabilities of ones in the stochastic bit streams A, B, and Y.

1.4 Stochastic Absolute Value Function

Li and Lilja [7] also developed a stochastic absolute value function. The state transition diagram is shown in Fig. 6. The output Y of this state machine is only determined by the current state S_i $(0 \leq i \leq N-1)$. If $0 \leq i < N/2$ and i is even, or $N/2 \leq i \leq N-1$ and i is odd, $Y = 1$; else $Y = 0$. The approximate function is,

$$y = |x|, \tag{3}$$

where x and y are the bipolar coding of P_X and P_Y. The proof of this function can be found in Li and Lilja [7].

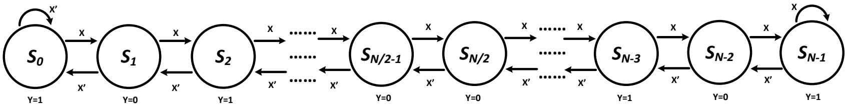


Fig. 6. State transition diagram of the FSM implementing the stochastic absolute value function

2 Case Studies

In this section, we demonstrate circuit constructs for common image processing tasks as case studies illustrating our method: image edge detection and median filter-based noise reduction [8].

2.1 Edge Detection

Classical methods of edge detection involve convolving the image with an operator (a 2-D filter), which is constructed to be sensitive to large gradients in the image while returning values of zero in uniform regions [8]. There are an extremely large number of edge detection operators available, each designed to be sensitive to certain types of edges. Most of these operators can be efficiently implemented by the SCEs introduced in this paper. Here we consider only Robert's cross operator as shown in Fig. 7 as an example [8].

$$\underset{G_X}{\begin{array}{|c|c|}\hline +1 & 0 \\ \hline 0 & -1 \\ \hline \end{array}} \qquad \underset{G_Y}{\begin{array}{|c|c|}\hline 0 & +1 \\ \hline -1 & 0 \\ \hline \end{array}}$$

Fig. 7. Robert's cross operator for edge detection

This operator consists of a pair of 2×2 convolution kernels. One kernel is simply the other rotated by $90°$. An approximate magnitude is computed using: $G = |G_X| + |G_Y|$, i.e.,

$$s_{i,j} = \frac{1}{2}(|r_{i,j} - r_{i+1,j+1}| + |r_{i,j+1} - r_{i+1,j}|),$$

where $r_{i,j}$ is the pixel value at location (i,j) of the original image and $s_{i,j}$ is the pixel value at location (i,j) of the processed image. Note that the coefficient $\frac{1}{2}$ is used to scale $s_{i,j}$ to [0, 255], which is the range of the grayscale pixel value.

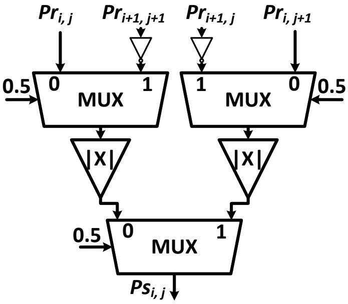


Fig. 8. The stochastic implementation of the Robert's cross operator based edge detection

The stochastic implementation of this algorithm is shown in Fig. 8. $P_{r_{i,j}}$ is the probability of ones in the stochastic bit stream which is converted from $r_{i,j}$, i.e., $P_{r_{i,j}} = \frac{r_{i,j}}{256}$.

So are $P_{r_{i+1,j}}$, $P_{r_{i,j+1}}$, and $P_{r_{i+1,j+1}}$. Suppose that under the bipolar encoding, the values represented by the stochastic bit streams $P_{r_{i,j}}$, $P_{r_{i+1,j}}$, $P_{r_{i,j+1}}$, $P_{r_{i+1,j+1}}$, and $P_{s_{i,j}}$ are $a_{r_{i,j}}$, $a_{r_{i+1,j}}$, $a_{r_{i,j+1}}$, $a_{r_{i+1,j+1}}$, and $a_{s_{i,j}}$, respectively. Then, based on the circuit, we have

$$a_{s_{i,j}} = \frac{1}{4}(|a_{r_{i,j}} - a_{r_{i+1,j+1}}| + |a_{r_{i,j+1}} - a_{r_{i+1,j}}|).$$

Because $a_{s_{i,j}} = 2P_{s_{i,j}} - 1$ and $a_{r_{i,j}} = 2P_{r_{i,j}} - 1$ ($a_{r_{i+1,j}}$, $a_{r_{i,j+1}}$, $a_{r_{i+1,j+1}}$ are defined in the same way), we have

$$\begin{aligned} P_{s_{i,j}} &= \frac{1}{4}\left(\left|P_{r_{i,j}} - P_{r_{i+1,j+1}}\right| + \left|P_{r_{i,j+1}} - P_{r_{i+1,j}}\right|\right) + \frac{1}{2} \\ &= \frac{s_{i,j}}{512} + \frac{1}{2}. \end{aligned}$$

Thus, by counting the number of ones in the output bit stream, we can convert it back to $s_{i,j}$.

2.2 Noise Reduction Based on the Median Filter

The median filter replaces each pixel with the median of neighboring pixels. It is quite popular because, for certain types of random noise (such as salt-and-pepper noise), it provides excellent noise-reduction capabilities, with considerably less blurring than the linear smoothing filters of the similar size [8]. A hardware implementation of a 3 × 3 median filter based on a sorting network is shown in Fig. 9. Its basic unit is used to sort two inputs in ascending order. It can be implemented by a comparator in a conventional deterministic implementation.

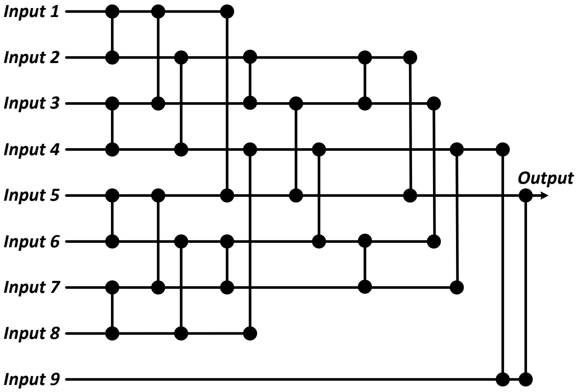


Fig. 9. Hardware implementation of the 3 × 3 median filter based on a sorting network

The stochastic implementation of this basic unit can be implemented by the stochastic comparator introduced in Section 1.3 with a few modifications. The circuit shown in Fig. 10 has the following functions:

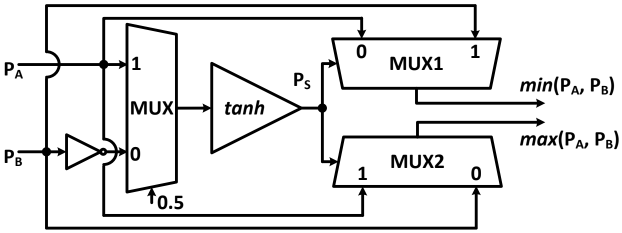


Fig. 10. The stochastic implementation of the basic sorting unit

- if $P_A > P_B$, $P_S \approx 1$, the probability of ones in the output of "$MUX1$" is P_B, which is the minimum of (P_A, P_B), and the probability of ones in the output of "$MUX2$" is P_A, which is the maximum of (P_A, P_B);
- if $P_A < P_B$, $P_S \approx 0$, the probability of ones in the output of "$MUX1$" is P_A, which is the minimum of (P_A, P_B), and the probability of ones in the output of "$MUX2$" is P_B, which is the maximum of (P_A, P_B);
- if $P_A = P_B$, $P_S \approx 0.5$, both the probabilities of ones in the outputs of $MUX1$ and $MUX2$ should be very close to $\frac{P_A+P_B}{2} = P_A = P_B$.

Based on this circuit, we can implement the sorting network shown in Fig. 9 stochastically.

3 Discussions and Conclusions

The stochastic paradigm offers a novel view of digital computation: instead of transforming definite inputs into definite outputs, circuits transform probability values into probability values; so, conceptually, real-valued probabilities are both the inputs and the outputs. The computation has a pseudo *analog* character, reminiscent of computations performed by physical systems such as electronics on continuously variable signals such as voltage. Here the variable signal is the probability of obtaining a one versus a zero in a stochastic yet digital bit stream. The circuits can be built from ordinary digital electronics such as CMOS. And yet they computed complex, continuous-valued transfer functions. Prior work has shown constructs for a variety of interesting functions. Most intriguing among these are the complex functions produced by linear finite-state machines: exponentiation, tanh, and absolute value.

Because a stochastic representation is uniform, with all bits weighted equally, it is highly tolerant of soft errors (i.e., bit flips). Computation on stochastic bit streams offers tunable precision: as the length of the stochastic bit stream increases, the precision of the value represented by it also increases. Thus, without hardware redesign, one has the flexibility to trade off precision and computation time. In contrast, with a conventional binary-radix implementation, when a higher precision is required, the underlying hardware must be redesigned.

A significant drawback of the paradigm is the long latency of the computations. The accuracy depends on the length of the bit streams; with long bit streams, each operation requires many clock cycles to complete. However, potentially the operations could be performed at a much faster clock rate, mitigating the latency issue.

The accuracy of the computation also depends on the quality of the randomness. If the stochastic bit streams are not statistically independent, the accuracy will drop. Furthermore, *correlation* is an issue in any circuit that has feedback or reconvergent paths. If the circuit has multiple outputs, these will have correlated probability values. In future work, we will study how to design circuits with multiple outputs – and so correlations in space. Also, we will study the impact of feedback – and so correlations in time.

Also, in future work we will study the *dynamic* behavior of stochastic constructs. We have observed that, using bit streams of length L to represent the inputs values, the output values of FSM-based stochastic constructs are always correct and stable after L clock cycles, no matter what the initial state. We will justify this claim mathematically. Finally, we will study a variety of FSM topologies, including 2D and 3D meshes, tori, and circulant graphs.

Acknowledgment. This work was supported in part by an NSF CAREER Award, No. 0845650.

References

1. Gaines, B.R.: Stochastic computing systems. Advances in Information System Science, Plenum 2(2), 37–172 (1969)
2. Brown, B.D., Card, H.C.: Stochastic neural computation I: Computational elements. IEEE Transactions on Computers 50(9), 891–905 (2001)
3. Qian, W., Riedel, M.: The synthesis of robust polynomial arithmetic with stochastic logic. In: 45th ACM/IEEE Design Automation Conference, DAC 2008, pp. 648–653 (2008)
4. Qian, W., Riedel, M.D., Rosenberg, I.: Uniform approximation and Bernstein polynomials with coefficients in the unit interval. European Journal of Combinatorics 32, 448–463 (2011)
5. Qian, W., Li, X., Riedel, M., Bazargan, K., Lilja, D.: An architecture for fault-tolerant computation with stochastic logic. IEEE Transactions on Computers 60(1), 93–105 (2010)
6. Li, P., Lilja, D.J.: A low power fault-tolerance architecture for the kernel density estimation based image segmentation algorithm. In: IEEE International Conference on Application - Specific Systems, Architectures and Processors, ASAP 2011 (2011)
7. Li, P., Lilja, D.J.: Using stochastic computing to implement digital image processing algorithms. In: 29th IEEE International Conference on Computer Design, ICCD 2011 (2011)
8. Gonzalez, R.C., Woods, R.E.: Digital image processing, 3rd edn. Prentice Hall (2008)

dRail: A Novel Physical Layout Methodology for Power Gated Circuits

Jatin N. Mistry[1], John Biggs[2], James Myers[2], Bashir M. Al-Hashimi[1],
and David Flynn[2]

[1] School of Electronics & Computer Science, University of Southampton, U.K.
{jnm106,bmah}@ecs.soton.ac.uk
[2] ARM Ltd., Cambridge, U.K.
{John.Biggs,James.Myers,David.Flynn}@arm.com

Abstract. In this paper we present a physical layout methodology, called dRail, to allow power gated and non-power gated cells to be placed next to each other. This is unlike traditional voltage area layout which separates cells to prevent shorting of power supplies leading to impact on area, routing and power. To implement dRail, a modified standard cell architecture and physical layout is proposed. The methodology is validated by implementing power gating on the data engine in an ARM® Cortex™-A5 processor using a 65nm library, and shows up to 38% reduction in area cost when compared to traditional voltage area layout.

Keywords: Physical Layout, Power Gating, Leakage.

1 Introduction

Leakage power can be as dominant as dynamic power below 65nm and poses a large source of power consumption in digital circuits [1]. A number of solutions have been proposed for reducing the leakage power dissipation of digital circuits which include the use of dual-threshold logic [2], application of reverse body bias [3] and power gating [4]. Power gating is proven to be the most effective and practical technique for reducing leakage power when logic is idle. For example, leakage power is lowered by 25x in the ARM926EJ-S™with the application of power gating [5]. In this technique, the parts of a digital circuit which are to be powered down are connected to the VDD power supply through a high threshold voltage (V_{th}) PMOS power gating transistor, which creates a pseudo supply, often referred to as a *virtual VDD* (VVDD), on the drain side of the power gating transistor. When the PMOS transistor is disabled, this virtual supply is disconnected from the true supply eliminating leakage currents in the power gated logic [5].

To facilitate the implementation of power gating in an ASIC, the logic to be power gated is grouped into a *voltage area* in the physical layout [5]. This is due to the inherent abutment that occurs between the power and ground connections of adjacently placed cells in a traditional standard cell library, which would otherwise cause the switched VVDD to be shorted with the always on

J.L. Ayala, D. Shang, and A. Yakovlev (Eds.): PATMOS 2012, LNCS 7606, pp. 245–255, 2013.

VDD in a power gating design. However, this physical separation has an area and routing cost on the design for a given performance target, as additional buffers and/or higher drive strength logic gates are inserted by the EDA tool to maintain performance [6] which also increases active power and will be shown in Section 3. Previous work has proposed to reduce the effects associated with the requirement for a voltage area by using distributed power gated rows [7,8]. The use of a custom standard cell library has also been proposed which allows two power supplies to be routed through each gate and duplicate gates are created for connecting to either of the supplies [9].

In this paper we propose a new physical layout design methodology, called dRail, which allows both power gated and non-power gated cells to be placed next to each other. This is unlike traditional voltage area layout [5] which separates power gated logic to prevent shorting of the switched and un-switched supplies. To achieve the dRail physical layout, first the standard cells are altered to stop them sharing the same power and ground supplies and prevents shorting of the switched and un-switched supplies without introducing additional cost to the standard cell architecture. Secondly, a modified cell layout is proposed to allow multiple supplies to be routed to every cell in the layout. The rest of this paper is organised as follows. Section 2 first describes the limitations of traditional voltage area layout before explaining the modified standard cell architecture and layout of the proposed dRail technique. Section 3 presents the validation of the proposed dRail methodology by implementation on a Cortex-A5. Section 4 concludes the paper.

2 Proposed Technique

The proposed dRail methodology allows both power gated and non-power gated cells to be placed next to each other to alleviate the need for a voltage area used in traditional power gating layout. dRail is achieved in two parts: firstly, the standard cell architecture is modified such that adjacent standard cells do not share the same power and ground by cutting back the power and ground connections. Secondly, the layout is modified to introduce a routing channel between site rows to allow both an always-on and switched supply to be routed to every cell.

Before the proposed dRail methodology is introduced, we first explain the limitations of the traditional voltage area layout used for power gated designs. Current standard cell gate libraries are designed such that the power (VDD) and ground (VSS) connections, usually in Metal1 (M1) (assumed for the rest of this paper), abut with adjacent cells when placed in a standard cell site row, Fig. 1. To ensure an uninterrupted VDD/VSS connection is available across the entire site row, any empty space is filled with M1 to create continuous M1 connections across the top and bottom of the site row which are referred to as *rails*. To prevent the always on VDD and switched VVDD supplies being shorted in the physical layout of a traditional power gated design, a voltage area is created [5] to separate the power gated logic from the always on logic denoted by the

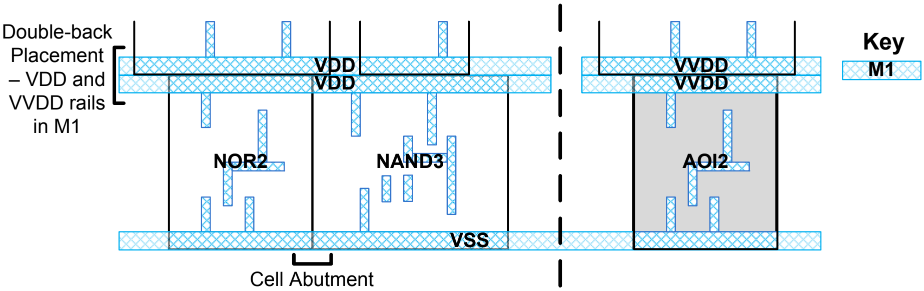


Fig. 1. Layout of power gating with traditional standard cell library and voltage area [5]. Note: break in the power supply rail and separation of shaded power gated cell

dashed line in Fig. 1. It should be noted that in this paper we assume a shared N and P well across voltage areas and a single switched VDD supply rail, as shown in Fig. 1, however a switched VSS supply rail is equally applicable. This separation can cause a greater distance between logically connected cells to arise which requires the addition of extra gates to maintain performance, resulting in area, routing and power overhead [6] (Section 3).

2.1 Modified Standard Cell Architecture

To overcome the requirement for a voltage area, and allow gates connecting to different power supplies to be placed adjacently, we propose to break, or 'de-rail', the continuous M1 rail across the standard cells to stop cells sharing the same power supplies, as in Fig. 1. To achieve this, we propose to shrink the power and ground (PG) pins of the standard cells so they no longer abut as shown in Fig. 2. Both the VDD and VSS connection are shrunk to allow the dRail technique to be used for switched VDD and/or VSS. Breaking the continuous M1 rail across the top and bottom of the site row means that each standard cell now has an independent VDD and VSS pin which can be connected to the necessary power supply, and will be shown in Section 2.2. The power gates are also modified as shown to enable them to be placed amongst the standard cells. To ensure the alterations shown in Fig. 2 do not introduce M1 spacing violations, the PG pins are cropped by $\frac{1}{2}$ the M1 design rule spacing from the edge. An added advantage of the proposed standard cell architecture is its versatility. The bounding box of the standard cell remains unchanged and therefore the cell occupies exactly the same area in placement. As the PG connections are only shrunk and the underlying function of the standard cell is unchanged, the cells can be used in a traditional placement flow by routing continuous M1 rails across the top and bottom of the cell rows with no change in power, performance or area.

2.2 dRail Layout

To demonstrate how the proposed modified standard cells, Section 2.1, can be used for a dRail layout, we convert the example shown in Fig. 1 with a single

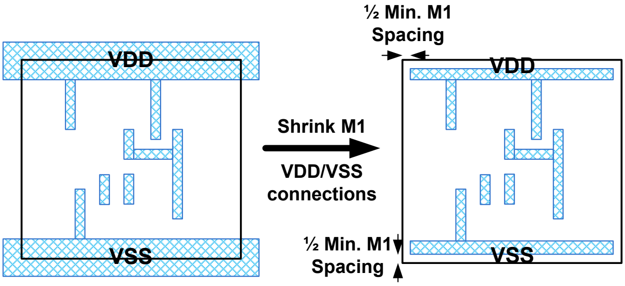


Fig. 2. Shrinking of VDD and VSS pins to stop power and ground abutment

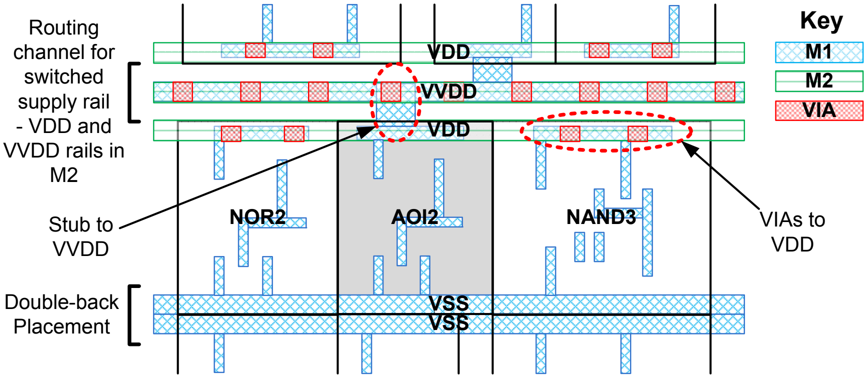


Fig. 3. Proposed dRail layout with a single switched supply rail, VVDD

switched VVDD supply into a dRail layout. The layout is shown in Fig. 3 and there are three key features. Firstly, unlike traditional voltage area layout where M1 is used to create a continuous VDD or VVDD rail across the top of the site row, Fig. 1, Metal2 (M2) is used to create a continuous VDD rail. This means that only cells that need to connect to this supply rail can be connected with a VIA between the rail and the VDD pin as demonstrated on the NOR2 and NAND3 gates in Fig. 3. Secondly, instead of traditional double-back placement as was shown in Fig. 1, a small routing channel is introduced between the site rows to accommodate the switched VVDD supply rail which is routed on both M1 and M2. This allows the AOI2 cell that had to be separated into a voltage area in Fig. 1, to now be placed adjacent to the always on cells and is connected to the VVDD with an M1 stub as shown in Fig. 3. It should be noted that in the implementation of dRail the N well is common to both the always-on and power gated cells which means the power gated cells are reverse body biased when they are shut down. Thirdly, in this example, the VSS supply is unswitched, so the rows are placed double-back for VSS and a continuous M1 rail is created to ensure an uninterrupted connection along the site row. The example given here is for a single switched rail, however, a switched VVDD and VVSS rail, such as is found in Zig-Zag power gating [8], can also be achieved with the same M2 and routing channel layout employed on both sides of the site row.

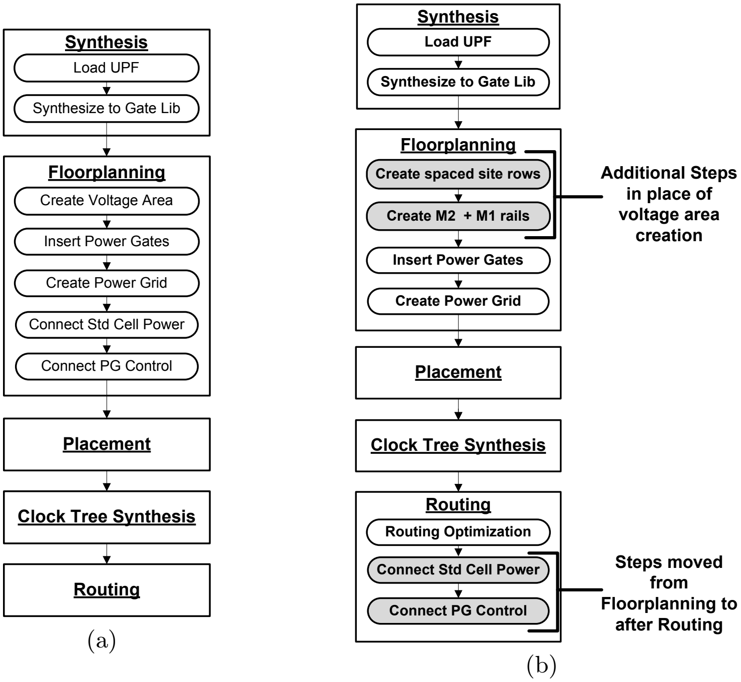


Fig. 4. Power gating physical design flow for (a) traditional voltage area (b) dRail

To achieve this layout small modifications are required to a power gating physical design flow using standard EDA tools. We assume the use of the IEEE1801 UPF standard, a leading power design intent standard for defining the strategy of a multi-voltage or power gated design [10]. The physical design flow of a power gated circuit using dRail is shown in Fig. 4(b) and shows some subtle differences to a traditional power gated design flow using a voltage area, Fig. 4(a), which are highlighted. The synthesis stage is unchanged, however, it must be noted that the UPF file used in the dRail physical design flow must define the power gates in the 'DEFAULT' global voltage area and not within power domains as would traditionally be done. This ensures the EDA tools do not expect the power gates to be placed inside a voltage area, which in a dRail layout do not exist. There are a number of changes in floorplanning with the most important exclusion being the creation of a voltage area. Instead, the site rows must be carefully positioned to create the routing channel seen in Fig. 3 and the M1 and M2 rails must be routed in the correct locations. Furthermore, since no voltage area is used in dRail, it is recommended that the power gates are placed in a grid pattern throughout the dRail physical layout as opposed to rings which can be used in a voltage area layout. These steps can be automated in the implementation scripts. Placement, clock tree synthesis and routing remain unchanged, but the connection of the standard cells to the power rails is postponed until after routing. This is because the location of the standard cells are not fixed until this

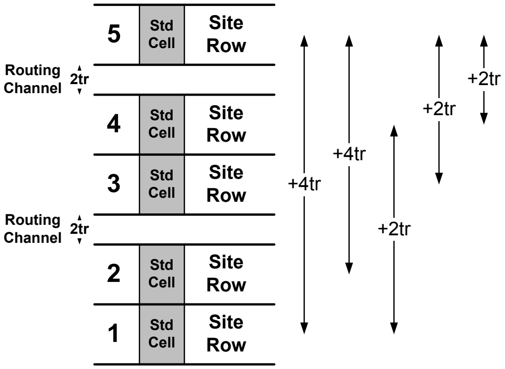


Fig. 5. Area overhead between standard cells (spreading)in dRail. tr = 1 Routing track

point and the stub and VIA connections required to connect the power to the modified standard cells (Fig. 3) would be incorrect had they been done earlier.

2.3 dRail Overheads

The proposed dRail design methodology introduces three overheads that must be considered in the physical layout. Firstly, the extra power routing done on M2 in the dRail layout, Fig. 3, creates routing blockage which can offset the routing improvements achievable with the dRail layout. Secondly, the additional routing channel between the site rows shown in Fig. 3 for inclusion of the switched supply rail results in 'dead' space as it cannot be used for placement. The area taken up by this additional routing space is the equivalent to one routing track per switched rail, per site row and is therefore dependent on the gate library being used. As an example, with a 12 track gate library i.e. each standard cell is 12 routing tracks in height, for a given number of site rows x, the loss of placement area for one additional power supply rail in dRail is $\frac{x}{12}$. Finally, the routing channel introduced between the site rows also results in spreading of the standard cells. For example, in the case with one switched rail, two standard cells placed directly opposite each other 3 site rows apart - e.g. rows 1 and 4 or 2 and 5 in Fig. 5 - results in the distance between them increasing by 2-4 routing tracks which can require the insertion of additional buffers to maintain performance. These overheads have an impact on the overall physical layout when using dRail, however bounded use of the dRail physical layout can minimise these overheads and improve the overall area and routing cost in a power gating physical layout and will be shown in Section 3.

3 Experimental Results

Three experiments were carried out to investigate the proposed dRail methodology. The first shows the impact of the overheads in the dRail layout methodology

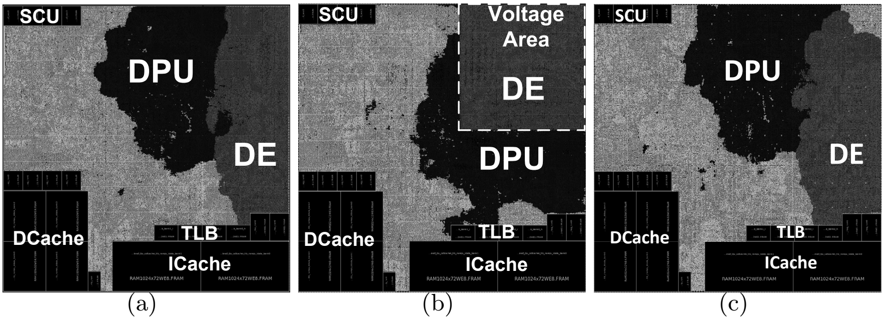


Fig. 6. Floorplan of A5 with interaction of Data Engine and Data Processing Unit (a) no power gating (b) DE power gated with voltage area (c) DE power gated with dRail

Table 1. Area, routing and power in no power gating and power gating with voltage area [5], and proposed dRail with difference to no power gating shown

	No Power Gating	**Voltage Area [5]**	**Diff (%)**	**Proposed dRail**	**Diff (%)**
Total Cell Area (μm^2)	1,246,592	1,286,710	3.2	1,258,415	0.9
of which: DE Area (μm^2)	206,007	216,407	5	211,277	2.6
PG Area Cost (μm^2)	0	2180	-	85,326	-
Total Placement Area (μm^2)	1,246,592	1,288,890	3.4	1,348,422	8.2
Routing Length (μm)	6,819,157	7,329,862	7.5	6,783,361	-0.5
Normalised Active Power	1	1.08	-	1.01	-

described in Section 2.3. The second and third show how the dRail layout can be bounded to reduce the effect of the overheads and hence improve the area cost associated with traditional voltage area layout. The experiments were carried out by power gating the data engine (DE) (floating point unit plus NEON^{TM}unit) in an ARM Cortex-A5 processor as its close interaction and tightly coupled nature with the rest of the data processing unit (DPU) makes it difficult to power gate. The processor was synthesized using a TSMC 65LP ARM Artisan$^{®}$ library modified for use with dRail and consisted of a single core, 16k Level-1 data and instruction cache, TLB cache and snoop control unit cache (SCU). All implementations targeted and achieved the same clock frequency and were fully place and routed using a UPF driven power gating flow with the Synopsys EDA tools. To ensure comparison of results was fair, the placement of the caches and silicon core area (1245μm x 1244.2μm) was kept fixed in all implementations.

3.1 Effect of dRail Overheads

An implementation of the Cortex-A5 was created without power gating and served as the baseline area, routing length and active power for the power gating implementations. Its floorplan can be seen in Fig. 6(a), and in particular, notice how the DE and DPU closely interact. Conversely, a floorplan of the same Cortex-A5 with a traditional voltage area power gating layout [5] used for the DE can be seen in Fig. 6(b) with the voltage area in the top right corner. As can be seen, the DPU is 'pulled' towards the DE and is done to reduce the distance between logically connected gates and maintain performance, but the voltage area shows a clear boundary (or guard band) between the two which results in $2180\mu m^2$ of 'dead' area which we refer to as power gating area cost in Table 1. The separation of these gates consequently has a 3.2% cost in total cell area from the addition of extra and larger gates and can be seen in Table 1. When coupled with the power gating area cost, the voltage area layout results in in a 3.4% increase in total placement area with respect to no power gating, and a 7.5% increase in routing. The increase in cell area and routing length consequently have an impact on the active power of the design which increases by 8%.

Fig. 6(c) shows the floorplan of the Cortex-A5 when using dRail throughout the entire physical layout. Unlike traditional voltage area layout, using dRail gives the EDA tool the freedom to place the standard cells anywhere resulting in a similar tightly coupled layout as the design without power gating, Fig. 6(a). The increase in total and DE cell areas are subsequently lower when compared to using a voltage area layout (Table 1) but is still higher than no power gating because of the spreading that occurs in dRail and hence the addition of extra and larger gates. Interestingly, routing length is reduced even when compared to no power gating and can be explained by a reduction in routing congestion from the introduction of the routing channels. Furthermore, the reductions in cell area and routing results in active power becoming comparable to no power gating. However, the overheads discussed in Section 2.3 result in the blanket use of dRail amounting to poor total placement area results in this test case. This is because the placement area wasted from the inclusion of routing channels incurs a large power gating area cost of $85{,}326\mu m^2$. This results in a higher total placement area than the voltage area layout and shows how no consideration of the impact of the overheads can result in an overall negative effect in terms of placement area.

3.2 Bounded dRail

To reduce the dRail overheads, the versatility of the proposed dRail standard cell architecture can be exploited to create bounded dRail layouts rather than using it throughout the entire physical layout. An example of this is shown in Fig. 7(a) where the right of the floorplan has a dRail layout with VDD and VVDD available for placement of power gated and always on cells together, and on the left of the floorplan, a traditional placement is used with only VDD available to eliminate the dRail spreading area cost in this placement area. As can be seen, the DE is entirely enclosed in the dRail boundary but the availability of the VDD supply rail allows logic gates from the DPU to be 'pulled' into the boundary to reduce the distance between logically connected gates. This is unlike a voltage area layout where the boundary enforced is exclusive to only the DE cells,

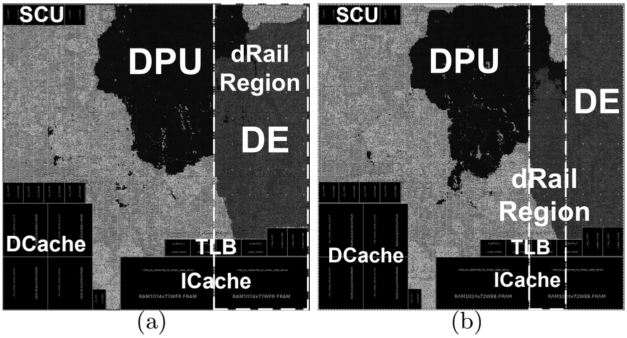


Fig. 7. Floorplan of A5 with interaction of Data Engine and Data Processing Unit (a) DE power gated with partial dRail (b) DE power gated with dRail on interface

Table 2. Area, routing and power in no power gating and power gating with voltage area [5], partial dRail, and interface dRail with difference to no power gating shown

	No Power Gating	**Voltage Area [5]**	**Diff (%)**	**Proposed Partial dRail**	**Diff (%)**	**Proposed Interface dRail**	**Diff (%)**
Total Cell Area (μm^2)	1,246,592	1,286,710	3.2	1,236,267	-0.8	1,236,528	-0.8
of which: DE Area (μm^2)	206,007	216,407	5	203,735	-1.1	203,587	-1.2
PG Area Cost (μm^2)	0	2180	-	35,752	-	18,359	-
Total Placement Area (μm^2)	1,246,592	1,288,890	3.4	1,294,167	3.8	1,278,623	2.6
Routing Length (μm)	6,819,157	7,329,862	7.5	6,574,952	-3.6	6,506,849	-4.6
Normalised Active Power	1	1.08	-	0.99	-	0.99	-

Fig. 6(b), and shows the strength of the proposed dRail layout methodology. Table 2 shows the results achieved with this bounded 'Partial dRail' implementation. As can be seen, bounded dRail improves upon the increase in total and DE cell area as well as routing length and power when compared to a voltage area layout but is also better than the blanket use of dRail throughout the layout (Table 1) because of a reduced impact from standard cell spreading overheads. The bounded use of dRail in this design has also helped to improve the power gating area cost compared to a blanket dRail implementation (Table 1). This brings the total placement area down to a comparable value to the voltage area layout whilst eliminating the 8% increase in active power.

An interesting thing to observe in Fig. 7(a) is that the interaction of the DPU and DE is largely isolated to the boundary. For this reason a second bounded implementation was created, Fig. 7(b), where dRail is only used on the interface of the two blocks to further minimise area overhead incurred in the DE region. The far right of the floorplan uses traditional placement with only VVDD for DE standard cells, and the left of the floorplan has only VDD for always-on standard cells. The results from this 'Interface dRail' physical layout are shown in Table

2. As can be seen, the area, routing and power are very similar to the 'Partial dRail' implementation but the power gating area cost has been reduced further. In this case an improvement of 38% is achieved over the voltage area layout when comparing the total placement area, whilst simultaneously eliminating the 8% increase in active power. These bounded dRail implementations demonstrate the versatility of the proposed methodology and shows how many power domains could be interleaved using the 'interface dRail' approach. Similarly, although one switched power rail and one power domain is shown in this test case, dRail with bounded placement has the potential for multiple switched rails for multiple power domains such as Zig-Zag power gating [8] or SoC interconnect.

4 Conclusion

This paper has proposed a new physical layout methodology, called dRail, for reducing the area, routing and power cost associated with using a voltage area in power gated designs by enabling power gated and non-power gated cells to be placed adjacent to one another. This is unlike traditional power gating layout where the standard cells are separated into a voltage area to prevent shorting of the switched and un-switched supplies. Experimental results on an ARM Cortex-A5 showed that bounded use of dRail can provide the largest improvements in area, routing and power whilst meeting the same performance target. The dRail methodology proposed in this paper is targeted at power gating in designs with highly interleaving logic such as zig-zag power gating or power gating in SoC fabric and builds on the multi-voltage EDA tools and flows with it being fully compatible with standard UPF power intent. dRail also has the potential for use in multi-VDD layout, but requires careful consideration of the back/forward biasing that could occur.

References

1. Agarwal, A., Mukhopadhyay, S., Raychowdhury, A., Roy, K., Kim, C.H.: Leakage Power Analysis and Reduction for Nanoscale Circuits. IEEE Micro. 26, 68–80 (2006)
2. Wei, L., Chen, Z., Roy, K., Johnson, M.C., Ye, Y., De, V.K.: Design and Optimization of Dual Threshold Circuits for Low-Voltage Low Power Applications. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 7, 16–24 (1999)
3. Tschanz, J.W., Narendra, S.G., Ye, Y., Bloechel, B.A., Borkar, S., De, V.: Dynamic Sleep Transistor and Body Bias for Active Leakage Power Control of Microprocessors. IEEE Journal of Solid-State Circuits 38, 1838–1845 (2000)
4. Mutoh, S., Douseki, T., Matsuya, Y., Aoko, T., Shigematsu, S., Yamada, J.: 1-V Power Supply High-Speed Digital Circuit Technology with Multithreshold-Voltage CMOS. IEEE Journal of Solid-State Circuits 30, 847–854 (1995)
5. Keating, M., Flynn, D., Aitken, R., Gibbons, A., Shi, K.: Low Power Methodology Manual. Springer (2007)
6. Weste, N.H., Harris, D.M.: CMOS VLSI Design: A Circuits and Systems Perspective, 4th edn. Addison-Wesley (2011)

7. Sathanur, A., Benini, L., Macii, A., Macii, E., Poncino, M.: Row-Based Power-Gating: A Novel Sleep Transistor Insertion Methodology for Leakage Power Optimization in Nanometer CMOS Circuits. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 19, 469–482 (2011)
8. Shin, Y., Paik, S., Kim, H.: Semicustom Design of Zigzag Power-Gated Circuits in Standard Cell Elements. IEEE Transactions On Computer-Aided Design of Integrated Circuits and Systems 28, 327–339 (2009)
9. Yeh, C., Kang, Y.: Cell-Based Layout Techniques Supporting Gate-Level Voltage Scaling for Low Power. IEEE Transactions On Very Large Scale Integration (VLSI) Systems 9, 983–986 (2001)
10. IEEE1801 Standard, http://standards.ieee.org/findstds/standard/1801-2009.html

Author Index