POLITECNICO DI TORINO

SCUOLA DI DOTTORATO

PhD in Electronics and Communications Engineering – XXVI cycle

PhD Dissertation

Robust and Efficient Globally-Asynchronous Locally-Synchronous (GALS) digital design

Marco Cannizzaro

Tutor
prof. Luciano Lavagno

Gennaio 2014

Summary

Process and operating condition variability creates a huge problem for current and future digital integrated circuits, because it forces them to operate at a speed, voltage and hence power and energy consumption which is very far from the optimum. System on Chip (SoC) architectures are born to meet some of the microelectronic trends. A single integrated chip contains an entire system with one or more central processors, several other chip-set, memories, and interfaces. The bottleneck of this approach are the interconnections between the various components. A global asynchronous communication is particularly suitable for this purpose because it removes most of the variable delays of the synchronous operation. At the same time, there is always the need to optimize the speed or power consumption of the computation and the Razor approach has been built for this purpose (trying to go below the synchronous safe operation).

So the goal of my work was to implement a new Globally-Asynchronous Locally-Synchronous (GALS) architecture that combines Safe Razor modules connected by flexible asynchronous communication channels (see Figure 1). In such architecture, both computation and communication are executed without the margins required by the synchronous worst-case methodology achieving better performance.

The thesis makes two contributions to the state of the art:

1. Safe-Razor: a metastability-robust adaptive clocking inside each synchronous GALS module.

2. M-of-N PID code: an efficient Delay-Insensitive (DI) protocol for the asynchronous communication between the GALS modules.

The Safe Razor is built on an earlier synchronous Razor architecture which cannot avoid potentially unrecoverable errors due to metastability. The synchronous Razor also requires an expensive restoring procedure available only on microprocessors or complex systems. The proposed Safe Razor confines both metastability and errors within a single module, without requiring any pipeline checkpointing

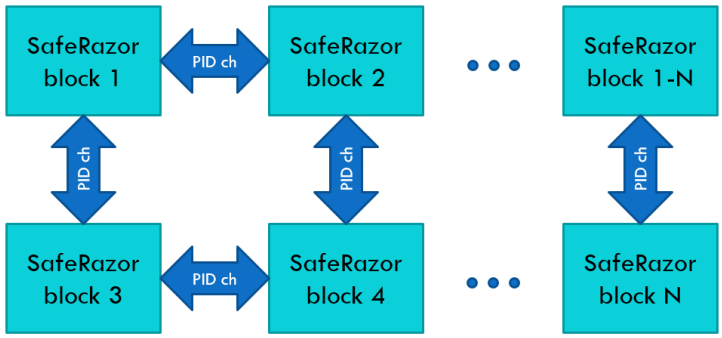

Figure 1. GALS architecture.

logic for restarting the system; it also ensure that both metastability and errors are always detected and corrected.

There is increasing interest in using m-of-n delay insensitive codes for robust asynchronous global communication, to support the design of coding-efficient and low-power channels. However, a fundamental obstacle in using these codes has been complex and expensive hardware support. This thesis addresses this issue, introducing the PID code which is a new delay insensitive (DI) data encoding protocol for better coding efficiency and simple hardware implementation.

Acknowledgements

I thank all professors and colleagues in the Electronics Engineering department of the Politecnico di Torino. I especially thank my advisor, prof. Luciano Lavagno, for the opportunity and honest advices, and Luca Necchi since I would have not been able to write the section 2.5 without his help. Thanks to Shlomi from the Technion-Israel Institute of Technology for the useful support in drafting section 2.1 and prof. Jordi Cortadella from the Universitat Politècnica de Catalunya for the cooperation in the Safe Razor project and careful timing analysis of section 2.3.
I also want to thank my colleagues of the microelectronics lab, Maurizio, Shahzad, Sarmad, Parinaz and Antonio, with whom I shared countless coffee breaks. I thank all collegues in lab4 of the Computer Engineering department for the lunches, the Friday Bar and the table football matches. I thank prof. Luca Carloni for giving me the opportunity to experience again an internship at the Columbia University in New York where I have gained new knowledge and established new friendships with Andrea, Christian, Giulia, Giuseppe, Johnnie, Paolo, Riccardo, Sara and finally PBR and my roommate Jen. Important role was played by the friends of the football matches on Friday morning and by my old friend Erika. A big thanks also to all my friends in Turin, Menfi and those around the world. I also thank my ex-roommates, Donato e Roberto, who hosted me while writing this thesis.

To my family and my girlfriend Donata finally go my warmest thanks.

****

Ringrazio tutti i docenti e colleghi del dipartimento di Ingegneria Elettronica del Politecnico di Torino. Ringrazio particolarmente il mio advisor, il prof. Luciano Lavagno, per le opportunità datemi e i sinceri consigli, e Luca Necchi senza il quale non sarei riuscito a scrivere il paragrafo 2.5. Grazie a Shlomi del Technion-Israel Institute of Technology per l'utile supporto nella stesura del paragrafo 2.1 e il prof. Jordi Cortadella dell'Universitat Politècnica de Catalunya per la collaborazione al progetto del Safe Razor e l'attenta timing analysis del paragrafo 2.3.
Voglio anche ringraziare i colleghi del laboratorio, Maurizio, Shahzad, Sarmad, Parinaz e Antonio, con i quali ho condiviso le innumerevoli pause caffé. Essenziali

sono stati anche i pranzi a mensa, i Friday Bar e le partite a calcio balilla con i colleghi del lab4 del dipartimento di Ingegneria Informatica. Ringrazio il prof. Luca Carloni per avermi permesso di rivivere l'esperienza alla Columbia University di New York dove ho acquisito nuove conoscenze e instaurato nuove amicizie con Andrea, Christian, Giulia, Giuseppe, Johnnie, Paolo, Riccardo, Sara e infine PBR e la coinquilina Jen. Ruolo importante anche ricoperto dagli amici del calcetto del venerdì mattina e gli amici ritrovati Berk ed Erika. Un doveroso ringraziamento anche a tutti gli amici di Torino, Menfi e a quelli sparsi per il mondo. Ringrazio anche i miei ex-coinquilini, Donato e Roberto, senza la cui ospitalità non avrei avuto in tetto sotto il quale scrivere questa tesi.

Alla mia famiglia e la mia fidanzata Donata va infine il grazie piu' grande.

Contents

Summary II

Acknowledgements IV

1 Introduction 1
- 1.1 Statement of the problem . . . 1
- 1.2 Asynchronous design benefits . . . 2
  - 1.2.1 Asynchronous design in communication . . . 2
  - 1.2.2 Asynchronous design in computation . . . 3
- 1.3 Description of the remaining chapters . . . 3

2 Safe Razor for computation 5
- 2.1 Metastability in Razor . . . 5
  - 2.1.1 Razor I . . . 6
  - 2.1.2 Razor II . . . 10
- 2.2 SafeRazor architecture in a GALS system . . . 12
  - 2.2.1 Normal mode of operation . . . 14
  - 2.2.2 Operation with metastability . . . 15
  - 2.2.3 Operation with error detection and correction . . . 16
- 2.3 Timing and Performance Analysis . . . 17
  - 2.3.1 Timing constraints . . . 17
  - 2.3.2 Normal mode of operation . . . 17
  - 2.3.3 Operation with metastability . . . 18
  - 2.3.4 Error detection . . . 19
  - 2.3.5 Error correction . . . 19
  - 2.3.6 Performance analysis: putting all together . . . 20
  - 2.3.7 Performance: an intuitive analysis . . . 21
- 2.4 2-phase Handshake Synchronization . . . 22
- 2.5 Results . . . 23

3 PID protocol for communication 27
3.1 Introduction . . . 27
3.1.1 Definition of terms . . . 27
3.1.2 Past work: M-of-N codes . . . 29
3.2 Motivation of the Proposed PID (Partial Inversion Data) code . . . 39
3.3 PID code implementation . . . 42
3.3.1 Example: the 2-of-7 PID code . . . 42
3.3.2 Hardware implementation of general M-of-N PID codes . . . 43
3.3.3 Decoder hardware of the M-of-N PID code . . . 44
3.3.4 Encoding algorithm for the M-of-N PID code . . . 45
3.4 Practical Completion Detection for 2-of-N codes . . . 45
3.4.1 Practical 2-of-N RZ CD . . . 46
3.4.2 Practical 2-of-N QDI RZ Completion Detection . . . 49
3.4.3 Practical 2-of-N NRZ Completion Detection . . . 50
3.5 Experimental Results . . . 56
3.5.1 Analytical results of the PID protocol . . . 56
3.5.2 Experimental results for 2-of-N Completion Detection . . . 57
3.5.3 Experimental results of a complete communication simulation 59

4 Conclusions and future works 73
4.1 Conclusions . . . 73
4.2 Future works . . . 74

List of Tables

2.1 Area comparison between traditional, Razor and SafeRazor circuits 26
3.1 Quasi-1-Hot 1-of-4 LETS code 34
3.2 Data Protocols Comparison table 37
3.3 m-of-n NRZ Transition-Signaling protocols Comparison table . . . 38
3.4 M-of-N PID codes from d=1 to d=7 43
3.5 Comparison table between Delay-Insensitive NRZ codes 68
3.6 Hardware evaluation: DIMS-style, basic[14] and proposed CD's (RZ and NRZ) 70
3.7 Area comparison between TE, LETS and PID codes in the communication system 70

List of Figures

1 GALS architecture. . . . iii
1.1 Synchronous margins for safe operation. . . . 1
1.2 Cycle period in asynchronous designs. . . . 1
2.1 Razor publications. . . . 5
2.2 Abstract of conceptual Razor I. . . . 6
2.3 Circuit introduced in [11]. . . . 7
2.4 Circuit introduced in [8]. . . . 7
2.5 Simulation of circuit of Figure2.4, when master becomes metastable and resolves before negative clock edge. Metastability and error detector signals shown. . . . 9
2.6 Simulation of circuit of Figure2.4, when master becomes metastable and resolves before negative clock edge. Flip-flop internal node signals are shown. . . . 9
2.7 Simulation of circuit of Figure2.4, when slave becomes metastable and resolves before positive clock edge. Metastability and error detector signals are shown. . . . 10
2.8 Simulation of circuit of Figure2.4, when slave becomes metastable and resolves before positive clock edge. Flip-flop internal node signals are shown. . . . 11
2.9 Razor II flip-flop diagram. . . . 12
2.10 Schematic of Razor II flip-lop path. . . . 12
2.11 Simulation of circuit of Figure2.9 (RazorII), error signal becomes metastable and resolves to low. . . . 13
2.12 A SafeRazor module in a GALS system. . . . 13
2.13 Timing diagram of the Safe Razor modes of operation. . . . 14
2.14 Metastability detection implementation . . . 15
2.15 A SafeRazor module in a 2-phase handshake system with multiple inputs/outputs. . . . 23
2.16 Error rate . . . 24
2.17 Average effective clock period . . . 25
3.1 NRZ protocol . . . 28

3.2 RZ protocol 28
3.3 1-of-4 RZ CD's (multi-field) 31
3.4 Basic architecture of a 2-of-9 RZ CD 31
3.5 1-of-4 NRZ CD's (multi-field) 34
3.6 2-of-9 Transition-Encoded decoder and completion detector 35
3.7 Optimized architecture a 2-of-9 NRZ CD 36
3.8 m-of-n NRZ Transition-Signaling protocols comparison chart . . . 39
3.9 Codeword format of the M-of-N PID code 40
3.10 Codeword format of the 2-of-7 PID code 42
3.11 Decoder architecture of the 2-of-7 PID code 42
3.12 Decoder hardware of the M-of-N PID code 44
3.13 Generic encoding algorithm for an M-of-N PID code 61
3.14 Specific encoding algorithm for the 2-of-7 PID code 62
3.15 Implementation of the RZ cell 63
3.16 Basic architecture of a 2-of-9 RZ CD 63
3.17 Proposed architecture of a QDI 2-of-9 RZ CD 64
3.18 Proposed architecture of the 2-of-9 NRZ CD 65
3.19 Implementation of the NRZ cell 66
3.20 Optimized architecture a 2-of-9 NRZ CD 67
3.21 Power and coding efficiency comparison 69
3.22 Decoder area overhead comparison 69
3.23 Decoder logical delay overhead comparison 70
3.24 Architecture of a 4 data bits communication channel 71
3.25 Performance chart between TE, LETS and PID codes in the communication system 71
3.26 Power chart between TE, LETS and PID codes in the communication system 71
4.1 Network on Chip architecture with Safe Razor modules connected by PID channels. 74

Chapter 1

Introduction

1.1 Statement of the problem

In the last years, in microelectronics field, we have seen a continuous increment in the complexity of integrated circuits. More and more functionality is implemented in a single chip due to continual scaling of transistors and increasing the size of integrated circuits. In addition to the complexity, there is also an increment in the performance of digital integrated circuits. Not only the continuous increment of the clock frequency (now above the Giga Hertz), but also the presence of independent computing nodes within the same chip allow to obtain higher performance.

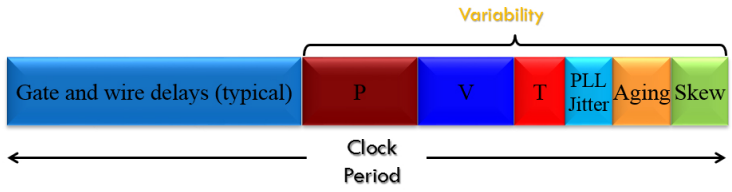

Figure 1.1. Synchronous margins for safe operation.

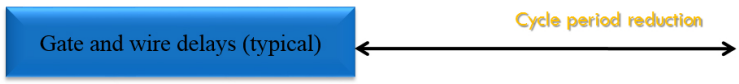

Figure 1.2. Cycle period in asynchronous designs.

In synchronous circuits, the clock period is chosen according to both the typical delays of logic gates and connections (the blue component to the left in Figure 1.1), and other variables that may alter the propagation times of the signals (such as the production process , temperature, supply voltage, and others). In asynchronous circuits (Figure 1.2), the time required by the logic gates increases since we need to add logic for encoding and decoding the channel, but this saves a lot on the variable components because while in synchronous circuits they are over-estimated, in the asynchronous circuits, the delays are the real ones.

1.2 Asynchronous design benefits

Synchronous Systems use a global clock as a centralized control signal which forces all components in the circuit to operate at fixed-rate. On the other hand, asynchronous circuits do not have a global clock and all components can operate at varying rate.

Asynchronous Designs have three potential main advantages:

1. *Lower power.* Since no global clock is required, components consume power only *on-demand*. Moreover, avoiding the global clock distribution, we also avoid that clock inherently consumes power continuously.

2. *Robustness and Scalability.* Components can operate at different speeds and the all design style becomes *object-oriented* with reusable components, flexible interfacing and scalable design. All of these features speed up the time-to-market.

3. *Higher Performance.* Because of the systems mismatching with central fixed-rate clock, they are not limited to worst-case clock rate. Moreover, the components interfaces provide instantaneous wake-up from standby mode.

Therefore, one promising recent direction is to design entirely asynchronous systems, or to combine synchronous cores with flexible asynchronous channels to form a global-asynchronous locally-synchronous (GALS) style system [23]. Both systems promise flexible and timing-robust communication, with low-power and ease-of-assembly.

1.2.1 Asynchronous design in communication

Coding theory is an approach to various science disciplines which adds features in data transmission methods. One concern of coding theory is designing codes that help synchronization. Considering two mechanism that require mutual synchronization, *asynchronous circuits* must explicitly generate sequence control signals

and they must encode/decode data to carrying information over self-timed on-chip interconnect links. Data encoding can add redundancy in the channel link and it adds the costs of encoding/decoding. In *synchronous circuits* much of this synchronization is implicit in the common clock signal.

To design robust asynchronous channels, *delay-insensitive (DI) codes* [24] are commonly used, where delays in receiving a data item, or even inter-bit timing skews, can be tolerated. However, one of the bottlenecks of commonly-used DI codes (i.e. dual-rail, 1-of-4) is that several have poor coding efficiency and high transition power. Alternatively, dense DI codes can be targeted, which are have high coding efficiency and low power (i.e. m-of-n). However, these codes typically have significant overheads in their supporting hardware, which often make their use infeasible.

1.2.2 Asynchronous design in computation

In order to optimize the speed or power consumption of the computation, the Razor systems (discussed in section 2.1) go below the safe operation (Figure 1.1). This obviously can lead to some errors which are detected and corrected by special circuits. When an error is detected, the operation is re-computed at a nominal safe mode. If errors do not occur so often, the system can achieve better performance. Razor has two drawbacks:

1. Razor is practical only with processors, because they already have the check-pointing and restarting logic required to re-start the system in case of errors.
2. Razor may have false positive error events in case of metastability.

An asynchronous approach can solve both the above issues.

1.3 Description of the remaining chapters

This thesis focuses on two different main aspects of a GALS architecture: the computation and the communication. Therefore the remaining chapters are divided accordingly:

Chapter 2: *Safe Razor for computation* This chapter starts with a background on the previous Razor architecture (Section 2.1) along with the motivation on why metastability can cause issues in the error detection (Section 2.1.1 for RazorI and Section 2.1.2 for RazorII). Safe Razor design is explained in detail in (Section 2.2) showing how it can be used to make the Razor idea works in case of metastability while improving its performance. Section presents a timing analysis of the Safe Razor GALS module. At the end, some experimental results are presented in (Section 2.5).

Chapter 3: ***PID protocol for communication*** A new class of Delay-Insinsitive code, *M-of-N Partial Invertion Data (PID)* code, is presented in this chapter. Section 3.1 gives an introduction on asynchronous data protocols and presents some of the most used DI codes. Section 3.3 presents the PID protocol, analyzes the codeword and the encoding/decoding process. Hardware implementations of a generic M-of-N Completion Detection (CD) is discussed in Section 3.4. Experimental results are presented in (Section 3.5).

Chapter 4: ***Conclusions and future work*** Conclusions (Section 4.1) and Future Works (Section 4.2) are presented in this chapter.

Chapter 2

Safe Razor for computation

2.1 Metastability in Razor

In this section we provide evidence and simulations that show the vulnerability of the original Razor approaches to metastability failures. Razor is often used to denominate the concept and several di erent derivatives of the main principle have been developed. Table2.1 summarizes some of the publications related to Razor designs. In [11] the Razor concept is was presented and it is usually denoted as Razor I. A modi ed version was later presented in [21] and is usually denoted as Razor II. Most of the other publications on razor circuits are derivation of either Razor I or Razor II. We follow by analyzing the metastability properties of both frameworks.

Figure 2.1. Razor publications.

Chapter 4

Conclusions and future works

4.1 Conclusions

This thesis presents techniques for developing a robust and efficient Globally-Asynchronous Locally-Synchronous (GALS) digital design. The work is organized in two main areas: the computation and the communication parts. Asynchronous design techniques are used in both areas.

A new Safe Razor architecture is built on the previous Razor design for the computation within a GALS module. It removes the need for expensive checkpointing logic so allows one to implement the nearly-critical clocking approach proposed by the Razor architecture to any logic block, not just to microprocessors with a restartable pipeline. It totally eliminates metastability failures, which still affect the classical Razor approach if the metastability detection logic itself (which in that case was synchronous, while in our case it is asynchronous) goes metastable.

A new Delay-Insensitive M-of-N protocol (where only M wires flip for each data transaction), called Partial Inversion Data (PID), is proposed for the communication channels between the GALS modules. It is a NRZ code, having significant power and throughput benefits with respect to Return-to-Zero (RZ) codes. It is Level-encoded, meaning that the decoding process simply uses the values of the codeword and it has a generic encoding algorithm and decoder implementation (that works for any M-of-N PID code). As hardware component, a M-of-N CD linearly scales in size based on the number of its inputs N and number of transitions M. It does not require external phase conversion circuits, nor does it require

a separate serialized reset phase. It does the job for any M-of-N code (not just PID).

4.2 Future works

Figure 4.1 shows the final goal of my research activities. A complete GALS Network on Chip (NoC) with heterogeneous Safe Razor modules (i.e. microprocessors, hardware accelerators, peripheral interfaces and so on) connected through channels using PID protocol.

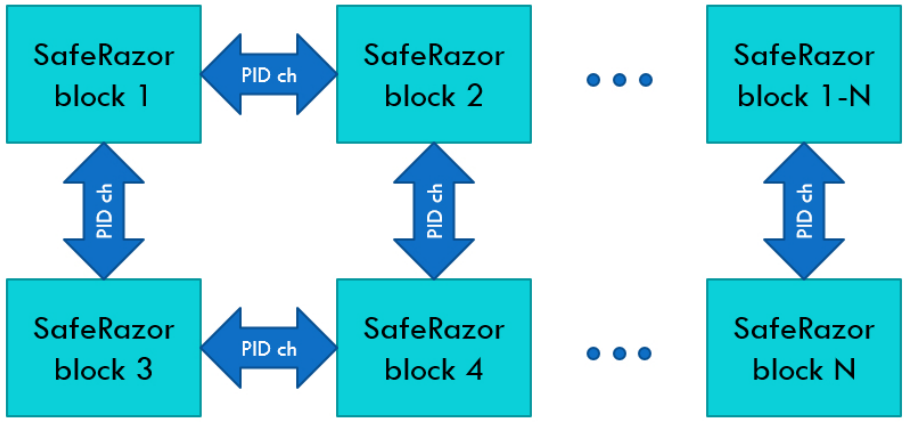

Figure 4.1. Network on Chip architecture with Safe Razor modules connected by PID channels.

This architecture should run faster than a classical synchronous NoC since it does not require to add margins for the computation and communication. Robustness is guaranteed in the computation by the asynchronous design of the Safe Razor which safely deals with metastability in the flip-flops; while in the communication the Delay-Insensitive characteristic of the PID protocol allows arbitrary delays and skew between bits in the channels.

Bibliography

[1] M.Y. Agyekum and S.M. Nowick. A delay-insensitive bus-invert code and hardware support for robust asynchronous global communication. *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*, pages 1–6, 2011.

[2] W. J. Bainbridge, W. B. Toms, D. A. Edwards, and S. B. Furber. Delay-insensitive, point-to-point interconnect using M-of-N codes. In *IEEE Async Symp.*, pages 132–140, May 2003.

[3] S. Beer, R. Ginosar, M. Priel, R. R. Dobkin, and A. Kolodny. The devolution of synchronizers. In *IEEE Async Symp.*, pages 94–103, 2010.

[4] J.M. Berger. A note on error detecting codes for asymmetric channels. In *Information and Control*, pages 68–73, March 1961.

[5] K. van Berkel. *Handshake Circuits: an Asynchronous Architecture for VLSI Programming*, volume 5 of *International Series on Parallel Computation*. Cambridge University Press, 1993.

[6] K. van Berkel, M. Josephs, and S.M. Nowick. Scanning the technology: Applications of asynchronous circuits. *Proceedings of the IEEE*, 87(2):223–233, February 1999.

[7] Marco Cannizzaro, Weiwei Jiang, and Steven M. Nowick. Practical completion detection for 2-of-n delay-insensitive codes. In *ICCD'10*, pages 151–158, 2010.

[8] D.; Seokwoo Lee; Pant S.; Blaauw D; Austin T.; Flautner K.; Mudge T. Das, S.; Roberts. A self-tuning dvs processor using delay-error detection and correction. *Solid-State Circuits, IEEE*, 41:792–804, 2006.

[9] I. David, R. Ginosar, and M. Yoeli. An efficient implementation of boolean functions as self-timed circuits. *IEEE Trans. on Comp.*, 41(1):2–11, 1992.

[10] M. E. Dean, T. E. Williams, and D. L. Dill. Efficient self-timing with level-encoded 2-phase dual-rail (LEDR). In *Proc. of UC Santa Cruz Conf. on Advanced Research in VLSI*, pages 55–70, 1991.

[11] D. Ernst, S. Nam Sung Kim, Das, S. Pant, R. Rao, Toan Pham, C. Zieslera, D. Blaauw, T. Austin, K. Flautner, and T. Mudge. Razor: a low-power pipeline based on circuit-level timing speculation. In *IEEE Int. Symp. Microarchitecture (MICRO-36)*, pages 7–18, 2003.

[12] K.M. Fant. *Logically Determined Design*. Wiley Interscience, Hoboken, NJ, 2005.

[13] C. Jeong and S.M. Nowick. Optimization of robust asynchronous circuits by local input completion relaxation. In *ASP-DAC*, pages 622–627, January 2007.

[14] C. Jeong and S.M. Nowick. Optimization for timing-robust asynchronous circuits based on eager evaluation. In *IEEE Async Symp.*, April 2008.

[15] M.M. Khan, D.R. Lester, L.A. Plana, A. Rast, X. Jin, E. Painkras, and S.B. Furber. SpiNNaker: mapping neural networks onto a massively-parallel chip multiprocessor. In *International Joint Conference on Neural Networks (IJCNN)*, pages 2850–2857, 2008.

[16] A.J. Martin. Asynchronous datapaths and the design of an asynchronous adder. *Formal Methods in System Design*, 1(1):119–137, July 1992.

[17] P.B. McGee, M.Y. Agyekum, M.A. Mohamed, and S.M. Nowick. A level-encoded transition signaling protocol for high-throughput asynchronous global communication. In *IEEE Async Symp.*, pages 116–127, April 2008.

[18] Carver Mead and Lynn Conway. *Introduction to VLSI Systems*. Addison-Wesley, Reading, MA, 1980.

[19] S.J. Piestrak. Membership test logic for delay-insensitive codes. In *IEEE Async Symp.*, pages 194–204, 1998.

[20] T. Chaney D. Zar S. Beer, J. Cox. Mtbf bounds for multistage synchronizers. *Proc. IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, 19:158–165, 2013.

[21] S.Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D.M. Bull, and D.T. Blaauw. RazorII: In situ error detection and correction for pvt and ser tolerance. In *IEEE J. Solid-State Circuits*, pages 32–48, 2009.

[22] M.R. Stan and W.P. Burleson. Bus-invert coding for low power i/o. *IEEE Trans. on VLSI*, 3:49–58, 1995.

[23] P. Teehan, M. Greenstreet, and G. Lemieux. A survey and taxonomy of GALS design styles. *Design & Test of Comps.*, 24(5):418–429, Sept.-Oct. 2007.

[24] T. Verhoeff. Delay-insensitive codes—an overview. *Distributed Computing*, 3(1):1–8, 1988.

[25] Y. Zhou, D. Sokolov, and A. Yakovlev. Cost-aware synthesis of asynchronous circuits based on partial acknowledgment. In *IEEE Int. Conf. on CAD*, November 2006.