# Dual-Sum Single-Carry Self-Timed Adder Designs

P. Balasubramanian and D.A. Edwards
*School of Computer Science, The University of Manchester,*
*Oxford Road, Manchester M13 9PL, United Kingdom.*
*E-mails: {padmanab, doug}@cs.man.ac.uk*

## Abstract

*This paper presents designs of self-timed dual-sum single-carry or dual-bit adder function blocks, constructed using commercially available synchronous library resources (standard cells) and validated using synchronous tools. Specifically, the proposed adder modules qualify as either quasi-delay-insensitive or speed-independent and satisfy Seitz's weak-indication timing constraints. The delay-insensitive version of the ripple carry adder topology has been used to analyze the designs. The indication (completion) is either made implicit in the topology (local indication) or considerably isolated from the actual data path (a new variant of global indication). The proposed adders are found to exhibit improved power and performance parameters, whilst being competitive in terms of area, in comparison with those pertaining to other self-timed logic realizations.*

## 1. Introduction

As future deep sub-micron technologies are characterized by parametric variations, self-timed (ST) design becomes an attractive solution. In the SIA's 2008 update on design technology [1], parameter uncertainty is projected to increase from a current figure of 10% to 25% by 2020. Design modularity is anticipated to increase to 55% by 2020, from a current 38%. ST[1] designs benefit as they inherently absorb the deviations of device characteristics and are highly modular. As existing commercial EDA tools support only synchronous circuits, asynchronous function blocks are realized using synchronous resources and validated using synchronous tools. This paper deals with adders designed in a robust asynchronous style and incorporates two main original contributions:

---

[1] The term self-timed (ST) is used to generalize quasi-delay-insensitive (QDI) or speed-independent (SI) notions in this paper.

- Proposal and design of different ST dual-sum single-carry (DSSC) adder modules using standard cells and C-elements functionality.

- Introduction of global indication feature for the data path.

Integer addition is an important part of digital computer systems. A study of the operations performed by an ARM processor's ALU revealed that additions constituted nearly 80% [2]. About 72% of the instructions of a prototype RISC machine resulted in addition/subtraction operations [3]. Henceforth, we restrict our attention to adder designs adhering to the 4-phase dual-rail (DR) handshake protocol, the robust and classic approach rooted in Muller's pioneering work [4], with isochronic fork assumptions (which is the weakest compromise to delay-insensitivity) [5] implicit in the designs. In a DR protocol, a signal $x$ is encoded into two rails as $x1$ and $x0$, where $x1$ represents a true-bit and $x0$ represents a false-bit. A logic one is denoted with $x1$ assigned a logic one and $x0$ a logic zero, while it is vice-versa for logic zero. In compliance with 4-phase handshaking, application of inputs alternates between spacer (all-zeroes) and data in every cycle. The DR encoding protocol is basically a delay-insensitive (DI) code [6] and is widely preferred for its simplicity and robustness. Many earlier ST adder designs have employed this encoding for only the carry logic and so they are not DI, neither are they robust.

## 2. Previous Work

A function block (here, adder) is the asynchronous equivalent of a digital combinatorial logic circuit. Function blocks can be strongly indicating or weakly indicating [8]. A strong-indication function block waits for all of its inputs to become valid (empty) before producing valid (empty) outputs, while a weak-indication function block does not. However, it delays

to produce at least one valid (empty) output till all its inputs have become valid (empty).

Among the different ST adder realizations that pertain to [7] – [10] and [14], the weakly indicating single-bit full adder design based on [14] was found to exhibit minimum data path and function block delay. This is mainly due to the fast carry propagation resulting from weak-indication of carry outputs. Since this work relies on utilizing synchronous cells for realizing robust ST designs, comparison with [12], or improvisations based on it is not possible, as they are founded on custom macros (proprietary NCL macros) made available as part of a cell library. The method of [13] can give rise to gate orphans and so it has not been considered. Gate orphans primarily stem from unacknowledged transitions on gate output nodes. Gate orphans may not necessarily be hazardous, but they can become critical to proper circuit operation. The adder realizations of [7] – [10] do not pose such problems. They also have no difficult timing assumptions.

In case of [11], for the worst scenario of all the false outputs of a function block evaluating to a logic high when valid input data has been applied, all the sum terms of the monotonic subnet DRN would have become enabled. When spacer is applied, even with a single sum term becoming disabled, and with ORN and CEN being reset, all the false outputs may evaluate to the correct empty state. This is a problematic situation, as transitions on the other intermediate gate (OR gate) output nodes would not be properly acknowledged, thereby giving room for creation of gate orphans. Hence, timing assumptions are necessary for proper operation and the design is not QDI or SI. Moreover, it suffers from high power dissipation, because all sum terms of the monotonic DRN become active for the worst case scenario resulting in high switching activity.

## 3. Proposed Dual-Bit ST Adder Modules

C-elements, which form the backbone of robust ST architectures, have been realized using standard library cells in our earlier work [14]. A C-element, also used as a latch, outputs logic high (low) only when all its inputs become logic high (low); otherwise they retain their existing state. They can be thought of as an AND gate for transitions. Hence, they are inherently DI and are basically strongly indicating. Figure 1 shows the gate level detail of a 3-input Muller C-element (CE3). In a slightly different fashion, a CE4 has also been realized; while a CE2 is realized in the standard manner by using a single AO222 cell.

The proposed designs of DSSC adder modules have been made technology-dependent with a focus on improving performance; nevertheless, they can be configured to remain technology-independent as well. This is feasible as all the larger input cubes can be decomposed into physically realizable smaller input cubes in a strict SI fashion without introduction of gate orphans. This advantage mainly stems from the design procedure that is proposed in this work.
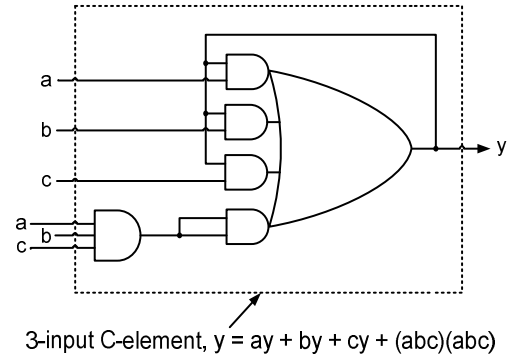


3-input C-element, $y = ay + by + cy + (abc)(abc)$

Figure 1. Cell based implementation of CE3

### 3.1. General design method

- Obtain the minimum sum-of-products (MSOP) expressions for all the DR function outputs.

- Translate the MSOP expressions into minimum disjoint SOP (MDSOP) formats.

- Perform SI decomposition on MDSOPs, in a target library aware fashion.

- Enable physical realization through technology mapping, preserving speed-independency.

MSOP equations for both the true and false function outputs can be obtained using a two-level logic minimizer: Espresso. A minimum mutually orthogonal SOP can be obtained from a MSOP by using the distributive, complementarity and absorption axioms of Boolean algebra. In such a format, every Boolean cube is mutually orthogonal to every other Boolean cube comprising it. Further information regarding these is available in [15]. Also, sufficient information about SI decomposition is elucidated using set theory in [15].

### 3.2. Adder realizations

A DSSC adder block consists of five inputs ($a1$, $a0$, $b1$, $b0$ and $cin$) and three outputs ($cout$, $sum1$ and $sum0$) represented in a DR encoded format, where ($a1$, $a0$) and ($b1$, $b0$) could represent the addend and augend inputs and $cin$ is the input carry. $cout$ is the overflow bit from the block and $sum1$ and $sum0$ are the most and least significant sum outputs respectively.

Three novel DSSC adder designs have been proposed in this work. In fact, they are three slightly but significantly different versions derived from one general design (at the theoretical level) and they give rise to structural differences in two ways even for the same topology adopted. The difference arises mainly with respect to the manner of indicating the arrival of all the primary function block inputs. It might be expected, that for a robust ST ripple carry adder (RCA) topology based on the DSSC adder, the time complexity would be reduced by half. But, due to the extra delay incurred in obtaining the least significant DR carry outputs; this is not achievable in practice.

Figure 2 shows the DSSC adder module realized with **C**-elements, **C**omplex gates, **A**ND gates and **O**R gates (referred to as DSSC_CCAO). If all the AND gates (which are highlighted in dashed lines) are replaced by C-elements and if the input completeness indication circuit synchronized with the least significant sum output logic is eliminated, then the resulting circuit would consist of only **C**-elements, **C**omplex gates and **O**R gates and so it shall be identified as DSSC_CCO. ISUM01 and ISUM00 nodes would then represent true and false least significant sum output bits respectively.

A 32-bit ST RCA constructed using DSSC_CCO or DSSC_CCAO modules is shown in figure 3. This is basically an adder encompassing local indication, as each individual adder module comprising it conforms to weak-indication constraints on their own. This is also due to the reason that a valid combinatorial circuit cascade of strong or weak-indication function blocks is itself a strong or weak-indication function block [8]. This property facilitates composing smaller blocks into a single large function block. The most significant DR sum outputs certainly indicate the arrival of all the DR augends and addends, and the DR input carry in cases; the least significant DR sum outputs definitely indicate the arrival of the DR input carry to this adder stage at all times. An alternative is to eliminate the input completeness indication circuit (excluding input carry) of figure 2 alone and use the remnant logic as the basic building block of the ST carry-propagate adder. In this situation, the weak-indication criteria may not be satisfied even within the module and so the overall circuit indicatability is to be taken care of by a separate logic (mainly meant to synchronize all the DR augends and addends), as portrayed in figure 4. Hence, the adder shown here is basically a modified version of DSSC_CCAO adder, with local indication no longer implicit in the actual data path but taken care of separately (globally) and hence is referred to as DSSC_CCAO_global adder.

## 4. Results and Conclusion

A weakly indicating traditional single-sum single-carry (SSSC) or single-bit adder based on [14] is referred to as proposed_SSSC in Tables 1 and 2. ST DSSC adder modules were realized based on [8], [9] and [10] as well and they are referred to as Seitz_DSSC, DIMS_DSSC and Toms_DSSC in the tabular columns. In fact, Toms_DSSC adder is alone strongly indicating. Design [7] is excessively large due to high logic duplication. In fact, an ST DSSC adder design based on [8] and [9] would not be realizable in practice, with most modern libraries. This is due to the requirement for AND gates with fan-in of 5 for [8] and the complication in realizing CE5 functionality with proper indication criteria in case of [9]. Besides, direct SI decomposition of [8] is not possible without avoiding gate orphans. So the method of [8] has been modified by introducing appropriate logic so that efficient SI decomposition became feasible. Nevertheless, direct SI decomposition of [9] is possible and hence it has been performed optimally. Further, performance oriented peephole optimizations were carried out on all the designs corresponding to the above approaches, to set the tone for a fair comparison.

Table 1. Area metric for different ST adders

| Realization style | Cells area ($\mu m^2$) |
|---|---|
| Proposed_SSSC [14] | 5924 |
| Seitz_DSSC [8] | 15514 |
| DIMS_DSSC [9] | 20484 |
| Toms_DSSC [10] | 9572 |
| DSSC_CCO [This work] | 13700 |
| DSSC_CCAO_local [This work] | 8788 |
| DSSC_CCAO_global [This work] | 8623 |

Tables 1 and 2 list the design metrics of the various adders, to realize a ST RCA of size 32 bits, on the basis of typical, worst and best case corners. All the adder's outputs have been uniformly configured with fanout-4 drive strength, while their inputs are configured with the driving capability of a minimum sized inverter in the library. Similar delay-optimized completion detection (CD) circuits were used for all the ST adders. Suitable buffer cells were provided within all the adder modules mainly to avoid timing violations, that results owing to a single acknowledge input feeding all the adder outputs in every stage of the cascade. However, for the DIMS_DSSC adder, in case of typical and best case corners, the OR gates generating the DR carry output have somewhat higher loading capability. This provision has been made to eliminate timing violations.
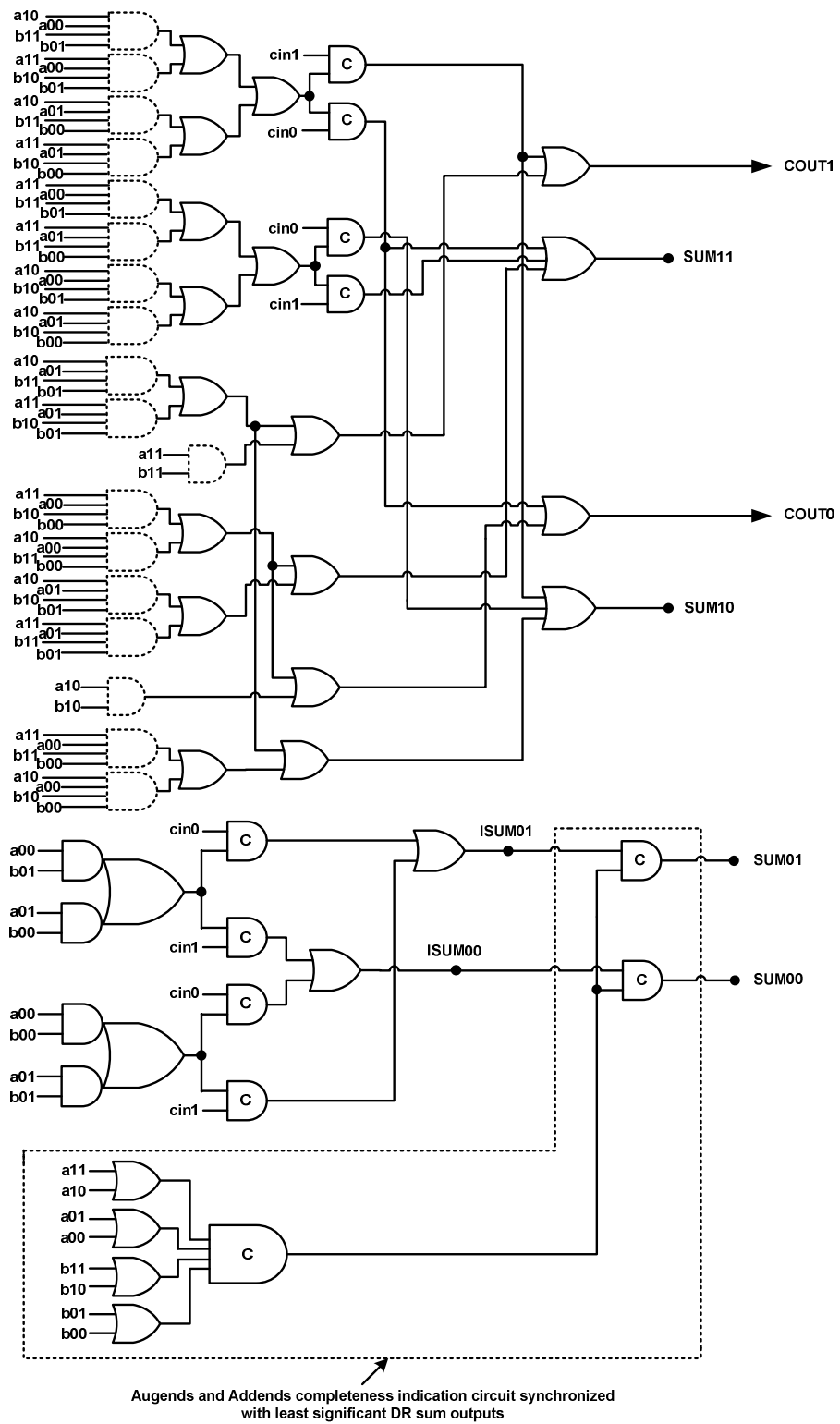
Augends and Addends completeness indication circuit synchronized
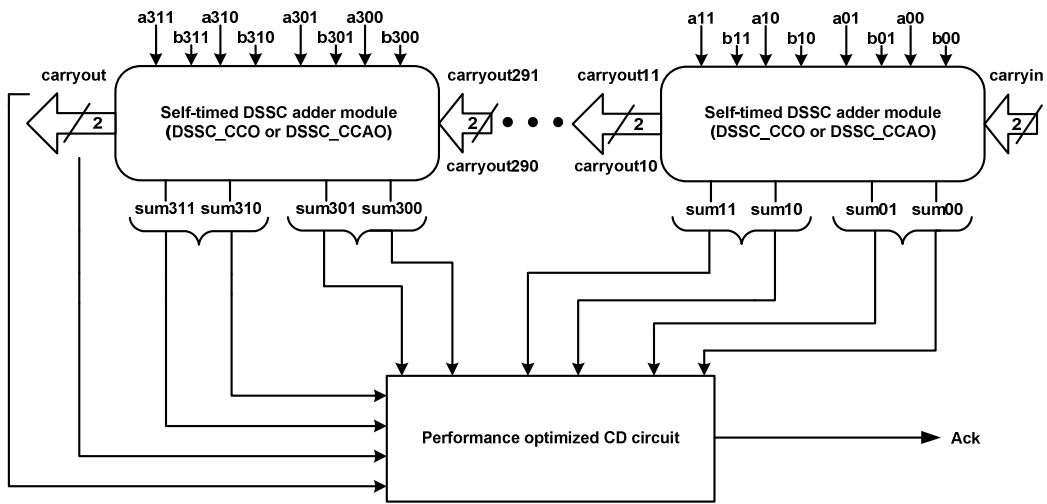with least significant DR sum outputs

Figure 2. ST DSSC_CCAO adder module

Figure 3. 32-bit DSSC ST adder based on DI RCA topology featuring local indication
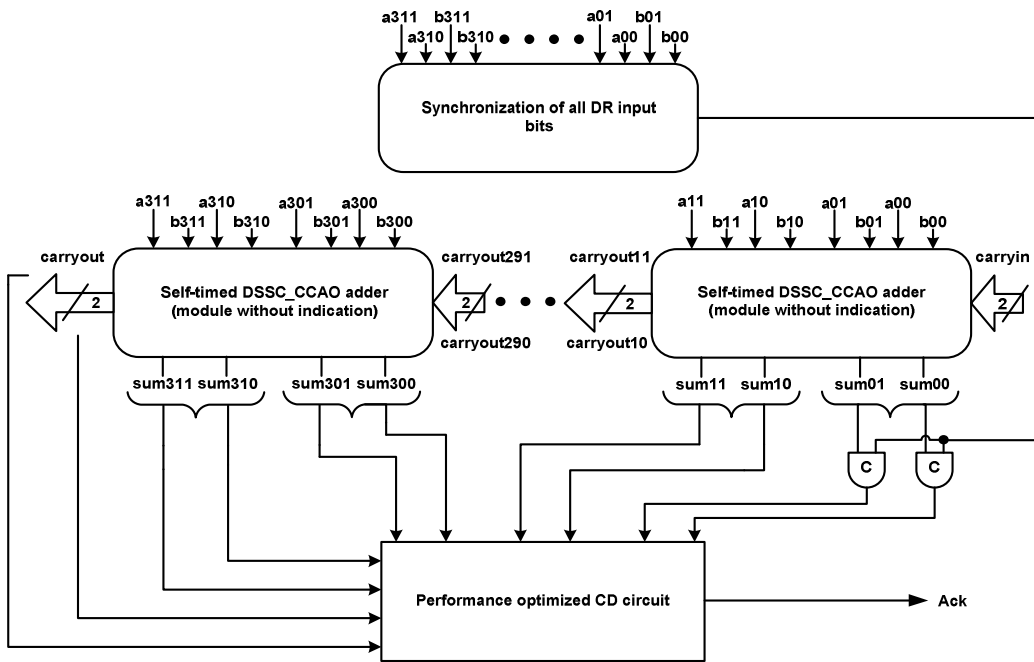(constructed using DSSC_CCO or DSSC_CCAO adder modules)



Figure 4. 32-bit DSSC ST adder based on DI RCA topology incorporating global indication
(constructed with a modified version of DSSC_CCAO adder module, shown in figure 2)

Functional simulation has been performed using NC-Verilog. PrimeTime and PrimeTime PX have been used to estimate delay, cells area and power figures respectively, inclusive of wire load information. Power results are derived, based on the application of input trace of *dc1* (a simple MCNC benchmark function) to all the ST adders. Inputs are applied every 20ns, 35ns and 15ns for the typical, worst and best case library specifications respectively. The simulations are all based on a virtual clock (not source), used only as a remote reference for guiding the application of inputs at a specific data rate, avoiding timing loops breaking during static timing analysis. In comparison with the DSSC_CCAO_global adder, the proposed_SSSC adder has reduced area occupancy by 31.3%.

Table 2. Delay and power metrics of ST adders

| PVT corner | Realization style | MDPD (ns) | FBD (ns) | TP (µW) |
|---|---|---|---|---|
| Typical (1.2V, 25°C) | Proposed_SSSC | 10.52 | 9.86 | 287.53 |
| | Seitz_DSSC | 19.05 | 18.38 | 355.65 |
| | DIMS_DSSC | 15.25 | 14.55 | 347.90 |
| | Toms_DSSC | 15.12 | 14.46 | 306.70 |
| | DSSC_CCO | 9.79 | 9.12 | 269.56 |
| | DSSC_CCAO_local | 9.37 | 8.69 | 310.05 |
| | DSSC_CCAO_global | 9.26 | 8.60 | 309.30 |
| Worst (1.08V, 125°C) | Proposed_SSSC | 18.10 | 16.94 | 131.04 |
| | Seitz_DSSC | 32.13 | 30.93 | 165.20 |
| | DIMS_DSSC | 33.47 | 32.21 | 163.31 |
| | Toms_DSSC | 25.65 | 24.48 | 142.03 |
| | DSSC_CCO | 16.79 | 15.57 | 125.23 |
| | DSSC_CCAO_local | 16.05 | 14.83 | 141.73 |
| | DSSC_CCAO_global | 15.88 | 14.69 | 141.58 |
| Best (1.32V, -40°C) | Proposed_SSSC | 6.89 | 6.46 | 479.18 |
| | Seitz_DSSC | 12.56 | 12.14 | 588.56 |
| | DIMS_DSSC | 10.09 | 9.64 | 576.05 |
| | Toms_DSSC | 9.96 | 9.54 | 504.54 |
| | DSSC_CCO | 6.43 | 6.00 | 450.75 |
| | DSSC_CCAO_local | 6.15 | 5.73 | 517.23 |
| | DSSC_CCAO_global | 6.09 | 5.66 | 515.71 |

In Table 2, acronyms MDPD, FBD and TP stand for maximum data path delay, function block delay and total average power dissipation respectively. MDPD is the summation of FBD and the delay of the CD circuitry, while total power is the gross of dynamic (switching + internal) and leakage power components.

From Table 2, we see that DSSC_CCAO_global has the lowest MDPD and FBD values across all the corners, while DSSC_CCO exhibits the least power dissipation in all the corners. Amongst the proposed adders, DSSC_CCAO_local features the highest power dissipation and lies between the other two dual-bit adders in terms of delay. The proposed_SSSC adder reports increase in delay over DSSC_CCAO_global adder by 13.6% and 14.7% with respect to MDPD and FBD respectively, on an average. In terms of total

average power, it suffers from increased power dissipation over DSSC_CCO adder by 6.7%, 4.6% and 6.3% across typical, worst and best case corners respectively. Excluding Seitz_DSSC (which reported a delay increase over its SSSC version), DIMS_DSSC and Toms_DSSC adders reported mean reduction in MDPD compared to their SSSC versions by 20.6% and 15.2% respectively. Nevertheless, the gain came at an area expense of 2.2× and 1.5× respectively. Overall, the proposed adders exhibit an improvement in delay and power, but remain competitive in terms of area.

## 5. Acknowledgment

## 6. References

[1] SIA's ITRS Report 2007, http://www.itrs.net

[2] J.D. Garside, "A CMOS VLSI implementation of an asynchronous ALU," *Proc. IFIP Working Conf. on Asynchronous Design Methodologies*, pp. 181-207, 1993.

[3] J.L. Hennesy and D.A. Patterson, *Computer Architecture: A Quantitative Approach*, Morgan Kaufmann, 1990.

[4] D.E. Muller, "Asynchronous logics and application to information processing," in *Switching Theory in Space Technology*, Stanford University Press, pp. 289-297, 1963.

[5] A.J. Martin, "Compiling communicating processes into delay-insensitive VLSI circuits," *Distributed Computing,* vol. 1, no. 4, pp. 226-234, 1986.

[6] T. Verhoeff, "Delay-insensitive codes: an overview," *Distributed Computing*, vol. 3, no. 1, pp. 1-8, 1988.

[7] N.P. Singh, "A design methodology for self-timed systems," *MIT CS Laboratory Tech. Report* TR-258, Feb. 1981.

[8] C.L. Seitz, "Chapter 7 – System Timing", in *Introduction to VLSI Systems*, Mead and Conway (Eds.), Addison-Wesley, '80.

[9] J. Sparso and J. Staunstrup, "Delay-insensitive multi-ring structures," *Integration, the VLSI journal*, vol. 15, no. 1, pp. 313-340, Oct. 1993.

[10] W.B. Toms, "Synthesis of Quasi-Delay-Insensitive Datapath Circuits," *PhD thesis*, University of Manchester, 2006.

[11] I. David, R. Ginosar and M. Yoeli, "An efficient implementation of Boolean functions as self-timed circuits," *IEEE Trans. on Computers*, vol. 41, no. 1, pp. 2-11, Jan. 1992.

[12] K.M. Fant and S.A. Brandt, "Null convention logic: a complete and consistent logic for asynchronous digital circuit synthesis," *Proc. IEEE ASAP*, pp. 261-273, 1996.

[13] X. Li and J.W. Sanders, "Efficient function block implementation of self-timed circuits," *Proc. 47th Midwest Symp. on Circuits and Systems*, vol. 2, pp. 269-272, 2004.

[14] P. Balasubramanian and D.A. Edwards, "A delay-efficient robust self-timed full adder," *Proc. 3rd IEEE Intl. Design and Test Workshop*, pp. 129-134, 2008.

[15] P. Balasubramanian and D.A. Edwards, "Power, delay and area efficient self-timed multiplexer and demultiplexer designs," *accepted for the 4th IEEE Intl. Conf. on Design and Technology of Integrated Systems in Nanoscale Era,* 2009.