Heterogeneously Encoded Dual-Bit Self-Timed Adder

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

***Abstract*—A novel heterogeneously encoded dual-bit self-timed adder design is presented in this paper. Heterogeneous encoding refers to a combination of at least two different delay-insensitive encoding schemes, adopted for the inputs and outputs. The primary motivation being that higher order 1-of-*n* encoding protocols facilitate reduction in terms of the circuit switching power dissipation compared to the basic dual-rail (1-of-2, which is the simplest 1-of-*n* code) encoding scheme. Here, *n* specifies the number of physical lines. The number of transitions gets reduced by O(*k*) over a dual-rail code, with *k* being the number of primary inputs and equals $\log_2 n$. The design of a dual-bit adder is considered to illustrate the advantage of the heterogeneous encoding scheme. The proposed adder design satisfies Seitz's weak-indication timing constraints. In comparison with dual-bit adders realized using other approaches, employing dual-rail encoding or heterogeneous encoding, the proposed design is found to be efficient in terms of delay, power consumption and area parameters.**

I. INTRODUCTION

The latest Semiconductor Industry Association's ITRS 2008 update highlights that design re-use (as a percentage of all logic) would increase from a current 38% to 55% by 2020. Over this period, parameter uncertainty (as a percentage effect on sign-off delay) is projected to increase from 10% to 25% [1]. Reliability has been labeled as one of the five cross-cutting design challenges, which drives home the point that design robustness is becoming a priority for deep submicron technologies. The above projections forecast and necessitate a considerable shift in the design paradigm from traditional synchronous to asynchronous (self-timed) logic, as the latter benefits in terms of greater modularity and ability to tolerate supply voltage, temperature and process parameter variations.

Self-timed (ST) logic design inherently guarantees that the requisite functionality is satisfied, irrespective of delays encountered in the design components or in the signal wires, as they are basically adaptive (elastic), especially when the signals are encoded in a delay-insensitive (DI) fashion. Though it has gathered interest as an attractive alternative to conventional digital logic design in the recent past, the fact that the vast majority of existing commercial EDA tools for design, verification and validation ideally target synchronous circuits cannot be overlooked. Therefore, in order to utilize the advantages offered by such sophisticated EDA tools and synchronous resources (standard cells), robust ST adder designs are realized using the gates of a standard cell library and also validated using industry-standard tools.

II. BACKGROUND, MOTIVATION AND PREVIOUS WORK

In this paper, we shall restrict our attention to function blocks (asynchronous equivalent of synchronous combinational logic), governed by the 4-phase handshake protocol with DI encoding adopted for inputs and outputs; the robust and classic approach rooted in Muller's pioneering work [2]. According to the 4-phase handshake protocol, input data alternates between valid data and empty data (an all zeroes state, also referred to as the spacer state) in every cycle.

We shall refer the adder by a function block in this article. Function blocks could satisfy strong-indication or weak-indication timing constraints: strongly indicating – if no outputs (spacer/valid) are produced until all inputs (spacer/valid) have arrived and weakly indicating – if some outputs (spacer/valid) could be produced based on even a subset of the inputs (spacer/valid). However, in the latter case, at least one output (spacer/valid) should not have been produced till all the inputs (spacer/valid) have arrived. The above criteria for strong and weak-indication circuits were formulated by Seitz in [3].

DR code is the simplest member of the general family of DI codes [8]. According to the DR encoding (DRE) scheme, a data bit *x* is encoded into two wires, namely *x*1 and *x*0, where *x*1 and *x*0 are called true and false bits respectively. A logic '1' is represented by *x*1 assigned a logic '1' and *x*0 assigned a logic '0', while a logic '0' is represented with the reverse assignment. The state of *x*1 and *x*0, both becoming '0' is referred to as the spacer and both *x*1 and *x*0 are not simultaneously allowed to transition to '1', as this is an illegal and invalid state. While DRE is used to represent only one bit of information, a 1-of-4 code, on the other hand, can be used to represent two non-redundant bits of information at a time by asserting only one of the four physical lines as logic high, as shown in Table 1. Similar to DRE, an all-zeroes state represents the spacer. It can be inferred from Table I, that for representation of two bits (say, *a* & *b*) of information, a 1-of-4 encoding approach would have only half as many transitions as that of a DRE approach. Hence, the former would suffer from reduced switching activity compared to the latter.

A major part of this research is supported by EPSRC, UK through the SEDATE project grant EP/D052238/1.

978-1-4244-3732-0/09/$25.00 ©2009 IEEE

Many ST logic design methods exist [3] - [7], [9]; nevertheless, their relative merits/de-merits have not been mentioned here for reasons of brevity. Among these, [7] is suitable for synthesis based on any arbitrary *m*-of-*n* encoding scheme while the others naturally lend themselves to DRE. The design procedures could be classified as quasi-delay-insensitive (QDI) or speed-independent (SI). A QDI model is basically a DI model, but with isochronic fork assumptions included [10]. In general, an isochronic fork allows a signal to travel to two different destinations whilst allowing for acknowledgement at only one end. Both QDI and SI models assume unbounded gate delays but place bounds on wire propagation delays, simplified by the above assumption.

TABLE I. DATA REPRESENTATION IN DUAL-RAIL AND 1-OF-4 ENCODING STYLES

Single-rail inputs	DRE	1-of-4 encoding
a b	***(a1 a0); (b1 b0)***	***(i0 i1 i2 i3)***
00	(01); (01)	(0001)
01	(01); (10)	(0010)
10	(10); (01)	(0100)
11	(10); (10)	(1000)

Since this work relies on utilizing synchronous cells for realizing robust ST designs, comparison with [11], or improvisations based on it is not possible, as they require the availability of proprietary NCL macros in a cell library. The adder realizations based on [3] – [7] and [9] are rather generic. They also have no difficult timing assumptions. The motivation for a dual-bit adder design stems from our recent work [13]. Compared to single-bit full adder realizations based on different methods [3] – [7], incorporating DRE, the dual-bit adder designs proposed in [13] – one based on C-elements, complex gates and OR gates (DSSC_CCO), another based on C-elements, complex gates, AND gates and OR gates with local indication (DSSC_CCAO_local) and the last design with global indication (DSSC_CCAO_global) were found to be beneficial in terms of delay and power, though there was an overhead in terms of area. Here, we consider heterogeneous encoding for both the inputs and outputs of the dual-bit adder.

III. HETEROGENEOUSLY ENCODED DUAL-BIT ADDER

As mentioned previously, the heterogeneous encoding (HE) scheme utilizes at least two different DI encoding schemes. In this work, we consider a dual-bit adder design which consists of five inputs (*a*1, *a*0, *b*1, *b*0 and *cin*) and three outputs (*cout*, *sum*1 and *sum*0), where (*a*1, *a*0) and (*b*1, *b*0) could represent the augend and addend inputs and *cin*, the input carry. Here, suffix '1' represents the most significant position and '0' represents the least significant position. *cout* is the overflow bit and *sum*1 and *sum*0 are the most significant and least significant sum output bits respectively.

The Boolean equations governing the adder are too big to detail here and so certain useful insights alone shall be mentioned in this paper. Minimum disjoint sum-of-products (MDSOP) expression is derived for all the adder outputs and SI logic decomposition and optimization is then performed. MDSOPs implicitly satisfy the monotonic cover constraint [12]. For information about SI decomposition, the reader is directed to [7]. The resulting proposed adder design is illustrated by figure 1. As can be seen in the figure, the augend and addend inputs and the sum outputs are 1-of-4 encoded – represented by (*a*0, *a*1, *a*2, *a*3), (*b*0, *b*1, *b*2, *b*3) and (*sum*0, *sum*1, *sum*2, *sum*3) respectively, while the input and output carries are encoded in DR format – represented by (*cin*0, *cin*1) and (*cout*0, *cout*1) respectively. The 1-of-4 encoded sum outputs indicate the arrival of all the adder inputs, while the 1-of-2 encoded carry output is not required do so. Hence, the output carry is weakly indicating and this property is favorable for facilitating faster propagation to the subsequent stage. But combined, the outputs certainly indicate the completion of computation within the adder block.

Since a valid combinatorial circuit cascade of strong or weak-indication function blocks is itself a strongly or weakly indicating function block [3], the individual adder modules could be cascaded to form an adder of arbitrary size (here, 32 bits). As far as robust ST adder topologies are concerned, many of the conventional high-performance synchronous adder topologies (such as carry look-ahead or parallel-prefix) cannot be translated into a straight-forward robust ST version, because the indication property cannot be satisfied and some would even lead to violation of the data encoding protocol (for e.g. a carry-select adder assuming input carry values). In fact, the ST ripple carry adder topology (shown in figure 2) would exhibit the least power dissipation and occupy the least area, as opposed to any other feasible ST adder topology and hence, a carry-ripple adder topology has been considered in this work to demonstrate the power savings that would result based on the proposed adder design, whilst featuring reduced delay. All the designs have been implemented based on the 130nm Faraday (UMC) bulk CMOS standard cell library. Muller C-elements, which form the backbone of robust ST architectures, have been realized using complex and ordinary gates, so that the indication (acknowledgement) property is preserved.

IV. SIMULATION MECHANISM, RESULTS & CONCLUSION

ST adders based on different approaches were realized with focus on delay-optimization. The dual-bit adder design based on [3] would be referred to as Seitz_DB_DRE. Dual-bit adders based on the delay-insensitive minterm synthesis (DIMS) approach [5] and that of [7], shall be identified as DIMS_DB_DRE and Toms_DB_DRE respectively. All these incorporate DRE. Our earlier dual-bit adder designs utilizing DRE shall be referred to as DSSC_CCO, DSSC_CCAO_local and DSSC_CCAO_global, as in [13]. The proposed design for dual-bit addition and that which realizes the same functionality based on [7], employing HE, shall be referred to as proposed_DB_HE and Toms_DB_HE respectively.

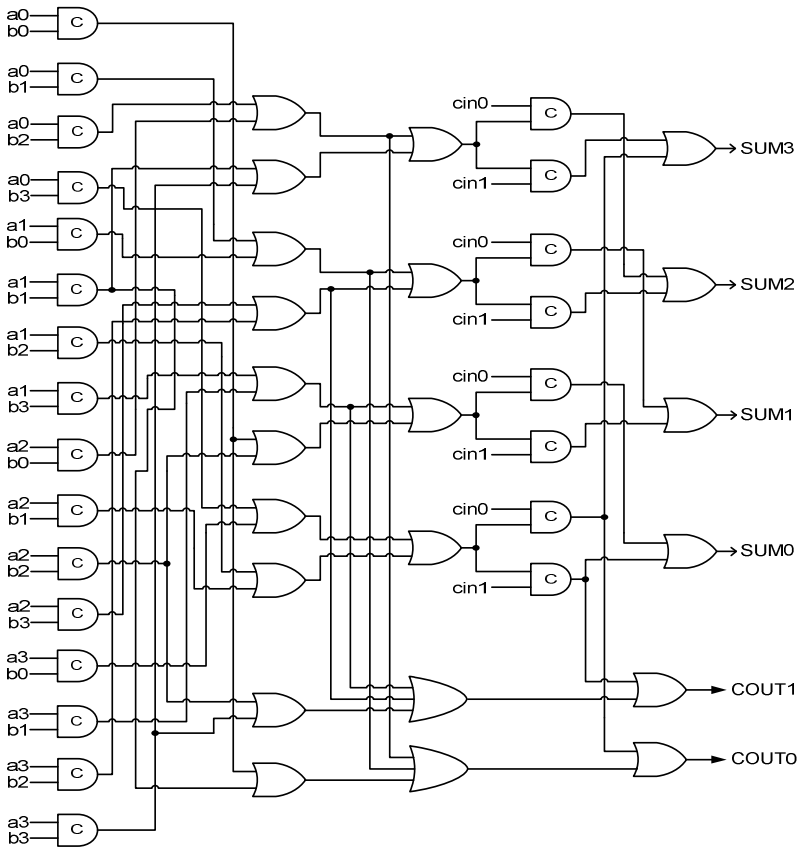

Figure 1. Proposed weakly-indicating heterogeneously encoded self-timed dual-bit adder module

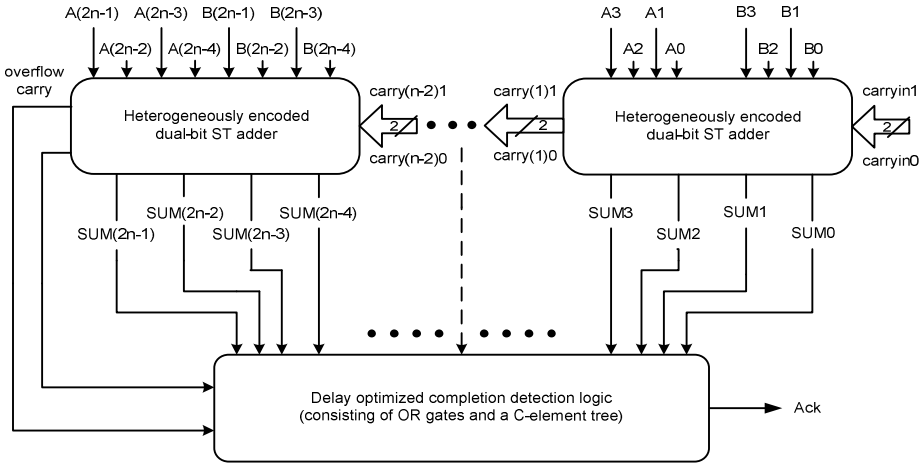

Figure 2. Self-timed version of the carry-ripple adder topology, based on dual-bit heterogeneously encoded adder modules

Notes: - The Muller C-element can be identified by the representation of an AND gate with the label 'C' in figure 1. It outputs logic high only when all its inputs are high and outputs logic low only when all its inputs are low. Otherwise, it retains its prior state. Adder modules incorporating DRE could be cascaded in a similar fashion, as shown in figure 2.

All the adders' outputs have been uniformly configured to exhibit fanout-4 drive strength, while their inputs are configured to possess the driving capability of a minimum sized inverter in the library. Appropriate minimum size buffer cells were provided within all the adder modules, mainly to eliminate timing violations. Though mostly minimum sized gates were used for the implementations, gate sizing was resorted to where required.

TABLE II. AREA OF VARIOUS 32-BIT ADDERS (130NM PROCESS)

Realization method	Cells area (μm^2)
Seitz_DB_DRE (Weak) [3]	15514
DIMS_DB_DRE (Weak) [5]	20484
Toms_DB_DRE (Strong) [7]	9572
DSSC_CCO_DRE (Weak) [13]	13700
DSSC_CCAO_local_DRE (Weak) [13]	8788
DSSC_CCAO_global_DRE (Weak) [13]	8623
Toms_DB_HE (Strong) [7]	8949
Proposed_DB_HE (Weak)	7717

TABLE III. DELAY AND POWER OF 32-BIT ADDERS (130NM PROCESS)

PVT corner	Realization method	MDPD (ns)	FBD (ns)	TP (µW)	SP (µW)
Typical (1.20V, 25°C)	Seitz_DB_DRE	19.05	18.38	355.65	226.47
	DIMS_DB_DRE	15.25	14.55	347.90	223.28
	Toms_DB_DRE	15.12	14.46	306.70	204.55
	CCO_DRE	9.79	9.12	269.56	155.69
	CCAO_local_DRE	9.37	8.69	310.05	180.21
	CCAO_global_DRE	9.26	8.60	309.30	177.20
	Toms_DB_HE	14.17	13.66	187.55	128.33
	Proposed_DB_HE	9.16	8.60	183.93	113.76
Worst (1.08V, 125°C)	Seitz_DB_DRE	32.13	30.93	165.20	100.46
	DIMS_DB_DRE	33.47	32.21	163.31	99.07
	Toms_DB_DRE	25.65	24.48	142.03	90.92
	CCO_DRE	16.79	15.57	125.23	69.07
	CCAO_local_DRE	16.05	14.83	141.73	80.07
	CCAO_global_DRE	15.88	14.69	141.58	78.67
	Toms_DB_HE	24.03	23.13	88.03	57.03
	Proposed_DB_HE	15.69	14.69	86.20	50.41
Best (1.32V, -40°C)	Seitz_DB_DRE	12.56	12.14	588.56	366.11
	DIMS_DB_DRE	10.09	9.64	576.05	362.11
	Toms_DB_DRE	9.96	9.54	504.54	330.73
	CCO_DRE	6.43	6.00	450.75	251.92
	CCAO_local_DRE	6.15	5.73	517.23	291.03
	CCAO_global_DRE	6.09	5.66	515.71	286.39
	Toms_DB_HE	9.35	9.03	308.18	207.37
	Proposed_DB_HE	6.02	5.65	302.81	184.10

Cadence NC-Verilog has been used for functional simulation and also to obtain the switching activity files for all the gate level simulations, while Synopsys PrimeTime and PrimeTime PX have been used for delay, cells area and power estimation respectively, including wire load information. A virtual clock (not source) has been used, only to act as a remote reference to guide the application of inputs to the ST adders at specific periods. Timing loops breaking has been avoided during timing analysis. The inputs to all the ST full adders correspond to that of a simple combinatorial benchmark, *dc1*, fed to the adders every 20ns, 35ns and 15ns for the typical, worst and best case specifications respectively.

Table II gives the area metric for the different adders. It can be observed that the proposed_DB_HE adder reports the least area occupancy amongst all other adders. In Table III, MDPD, FBD, TP and SP stand for maximum data path delay, function block delay, total average power and net switching power (which is indicative of the switching activity) respectively. FBD is the maximum path delay from the first adder stage to the last adder stage and MDPD is the sum of FBD and CD logic delay. TP is the sum of dynamic and static power components. Amongst all the ST adders, the proposed_DB_HE adder is found to consume the least power (both SP and TP) and also features the least delay metrics (MDPD as well as FBD). CCAO_global_DRE adder nearly equals the proposed_DB_HE adder in terms of speed, but in terms of TP and SP, it is expensive by 68.7% and 55.7% respectively. Although Toms_DB_HE adder is comparable to the proposed_DB_HE adder in terms of power dissipation, nevertheless it suffers delay increase to the tune of 54.4% with respect to MDPD and 58.7% with respect to FBD. Overall, it can be concluded that the proposed adder design is found to be optimal in terms of power, delay and area metrics.

REFERENCES

[1] SIA's ITRS reports on Design, Available: http://www.itrs.net

[2] D.E. Muller, "Asynchronous logics and application to information processing," Stanford University Press, pp. 289-297, 1963.

[3] C.L. Seitz, "Chapter 7 – System Timing", in *Introduction to VLSI Systems*, C.A. Mead and L.A. Conway (Eds.), Addison-Wesley, 1980.

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

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

[6] B. Folco et al., "Technology mapping for area optimized quasi-delay-insensitive circuits," *Proc. IFIP Intl. Conf. on VLSI*, pp. 55-69, 2005.

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

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

[9] I. David et al., "An efficient implementation of Boolean functions as self-timed circuits," *IEEE Trans. on Comp.,* 41(1), pp. 2-11, Jan. 1992.

[10] A.J. Martin, "The limitations to delay-insensitivity in asynchronous circuits," *Proc. 6th MIT Conf. on Adv. Res. in VLSI*, pp. 263-278, 1990.

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

[12] A. Kondratyev et al., "Hazard-free implementation of speed-independent circuits," *IEEE Trans. on CAD*, 17(9), pp. 749-771, 1998.

[13] P. Balasubramanian and D.A. Edwards, "Dual-sum single-carry self-timed adder designs," *accepted for IEEE ISVLSI*, 2009.