A Novel Pseudo 4-Phase Dual-Rail Asynchronous Protocol with Self-Reset Logic & Multiple-Reset

M.Santhi, S. Arun Kumar, G.S.Praveen Kalish, Siddharth Sarangan, G.Lakshminarayanan
Dept. of Electronics and Communication Engineering,
National Institute of Technology, Tiruchirappalli, Tamilnadu, India

ABSTRACT

This paper presents a novel pseudo 4-phase dual-rail protocol with self-reset logic suited for high speed asynchronous applications. The traditional 4-phase dual-rail requires the input to be of alternating valid and empty cycles. However the proposed pseudo 4-phase involves continuous stream of valid data without a separate empty cycle. The empty phase is generated internally so that the next valid data can be processed. Also self-reset logic for dual-rail protocol has been proposed in which the combinational blocks resets itself whenever its evaluation phase is completed and the data is latched at the pipeline register. The concept of multiple-reset i.e. resetting each of the gates in the combinational block between any two pipeline registers simultaneously has been introduced reducing the reset phase and hence increasing the throughput rate. An asynchronous 8-bit pipelined carry propagate adder was implemented in 0.18μm technology. The reset phase has reduced by 63.25% and 47.63% compared to the design without self and multiple reset for logic depth of three and two respectively. The results show that the reset phase varies inversely with the logic depth for the proposed design.

General Terms

Asynchronous Systems

Keywords

Asynchronous, dual-rail, pseudo 4-phase, self- reset, multiple-reset.

1. INTRODUCTION

Due to the recent trend towards multi-gigahertz systems, there are increasing design challenges to managing clock distribution. Asynchronous design, which replaces global clocking with local handshaking, has the potential to make high speed design more feasible. Since they are asynchronous, these pipelines avoid issues related to the distribution of a high-speed clock: e.g., wasteful clock power and management of clock skew. Moreover, the use of local handshaking, instead of global clocking, imparts the ability of localized flow control to the asynchronous pipeline: the number of data items in the pipeline is allowed to vary over time, depending on the operating speeds of the interfaces and stages. Thus, the proposed pipelines have a natural elasticity. In contrast, except for some recent approaches, traditional synchronous pipelines typically do not provide elasticity. Finally, the inherent flexibility of asynchronous components allows an asynchronous pipeline to interface with varied environments operating at different rates; thus, asynchronous pipeline styles are useful for the design of system on- a-chip (SoC) and reusable intellectual property (IP), and also have been shown to gracefully accommodate dynamic voltage scaling. Finally, the proposed control circuits are simple, small and fast, thus further minimizing the overhead of fine grain pipelining. Dual-rail is a commonly-used scheme to implement an asynchronous data path [1] [2]. In dual-rail, two wires (or rails) are used to implement each bit. The wires indicate both the value of the bit, and also its validity. In the 4-phase implementation, encodings of 01 and 10 correspond to valid data values 0 and 1, respectively. The encoding 00 indicates the reset or spacer state with no valid data and 11 is an unused (i.e., illegal) encoding. Encodings on the data path typically alternate between valid values and the reset state. Since the data path itself indicates the validity of each bit, dual-rail is effective in designing asynchronous data paths which are highly robust in the presence of arbitrary delays. Unlike in the 4-phase implementation, where definite codes represent definite values, in 2-phase, a new data is marked by a transition in the corresponding bit. A transition in false/true line indicates a zero/one respectively.

2. 2-PHASE AND 4-PHASE IMPLEMENTATION - A COMPARISON

A key limitation of LEDR is the difficulty of building efficient and simple robust function blocks [3]. There are two challenges: the complexity of handling the alternating encodings of odd and even phases and the overhead of enforcing hysteresis so that the output phase does not change until the phase of all inputs have changed. These requirements make LEDR functional blocks area and delay-inefficient, and hard to design without using custom transistor gates. Hence, in practice LEDR is rarely used to design function blocks.

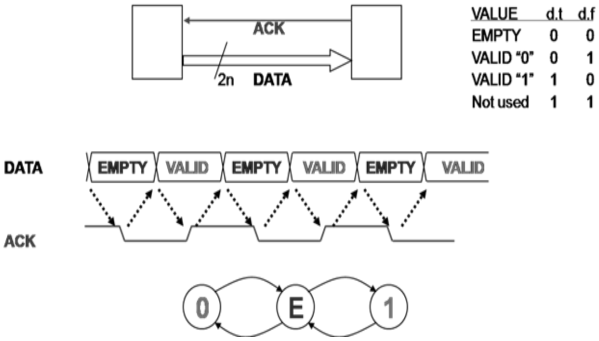

Figure.1 4-phase dual-rail protocol

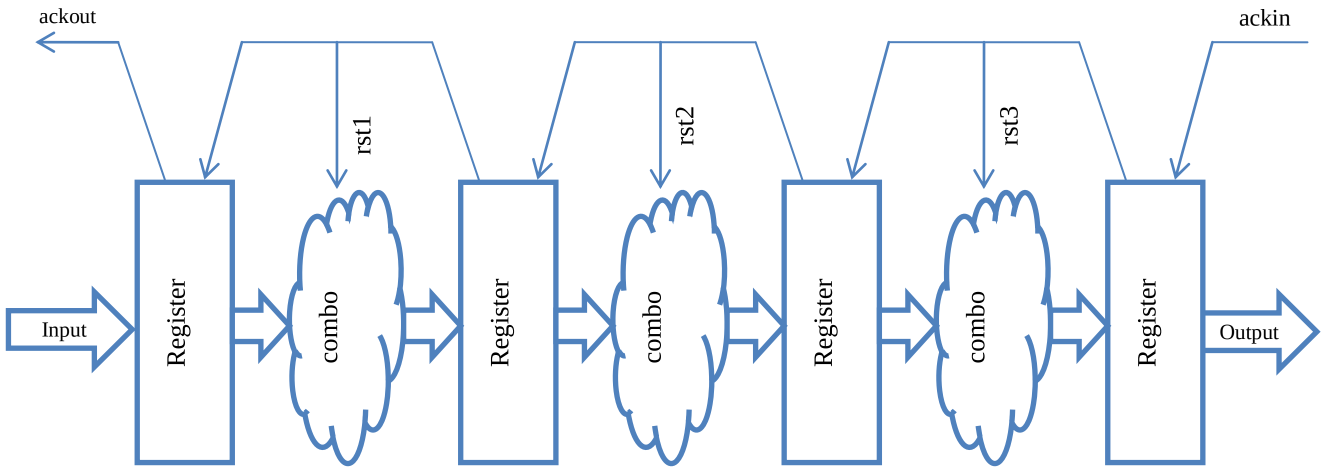

Fig.2 Block diagram of pseudo 4-phase dual-rail with self-reset

In contrast, 4-phase dual-rail is commonly-used and easy to implement functional blocks. But the 4-phase protocol as shown in Fig.1 requires alternate valid and empty data so that the block is reset after every computation and is ready to process the next data. The period of valid and empty phase is equal to the largest combinational path delay. This results in reduced throughput rate as the empty data takes time to propagate through the system.

3. DESIGN OVERVIEW

The main contribution of this paper is that it proposes a new dual-rail protocol that is optimized for low power and is simpler to build the system. Because of the inherent advantages and disadvantages of 2-phase (LEDR) and 4-phase protocols, we propose the pseudo 4-phase protocol. This protocol has similar features to that of 4-phase in the way that the acknowledge and the data signals used inside the system is 4-phase transitioning, but external to the system the regular empty phase of a phase system is not visible. So both at input and output we have streams of valid data. The empty phase is generated internally using self-reset logic that is described later. This results in lower number of transitions at the input and output due to non-return to zero signals as like the 2-phase. Unlike 2-phase, there is no explicit phase changes within the block thereby ensuring lower complexity of combinational blocks.

As shown in Fig. 2, each stage consists of a combinational and a reset network. Data flows from one stage to another through a latch in a linear pipeline. The time taken for the combinational block to process the data is the evaluation phase. As soon as the evaluation phase is over the combinational block is reset to accept the next data. An acknowledge signal is generated (a low to high transition) when a valid data is detected by the completion detection circuit. When the acknowledge signal is low, the combinational block is in evaluation phase. To insure proper data flow across stages, data is transferred from the current stage to the next one if the current stage is in the evaluate phase while the next stage is in the reset phase. The enable signal of the latch depends upon the acknowledge signal of the next stage as it denotes the commencement of reset phase in that stage. Hence, the latch separating both stages is enabled when both stages are in evaluate and reset phase respectively. Various forms of Self-Reset logic have been proposed in recent years. In globally self-resetting CMOS [4], the reset signal for each stage is generated by a separate timing chain which provides a parallel worst-case delay path. On the other hand, in locally self-resetting CMOS [5], the reset signal for each stage is generated by a mechanism internal to that stage. Most previous implementations of SRCMOS have required great care in device sizing in order to ensure that the reset pulses will occur at the correct times. This is especially difficult to achieve over a wide range of process, voltage and temperature variations.

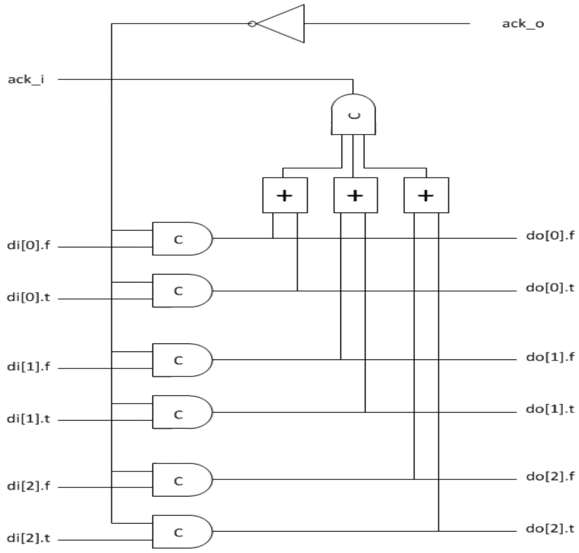

Fig. 3 Pipeline register with acknowledge generation

3.1 Self Reset Logic

The major drawback of the previous implementation of self-reset logic [6] is the usage of delay buffers that are required to delay the reset signal till a valid data is obtained at the end of the combinational block. While the inputs are traveling along the critical path of the combinational network, the reset signal is similarly traveling along the matching delay. This delay buffer delays the reset phase long enough to allow the outputs of the combinational network to stabilize. In the proposed Dual Rail self reset logic, the delay insensitive Muller C-element [7] is used in the completion detection circuit (as shown in the Fig. 3) to indicate the validity of all the bits of the data. This waits for all the bits to be valid eliminating the need to model the delay. Once all the bits become valid, acknowledge is generated as shown in the Fig.4.

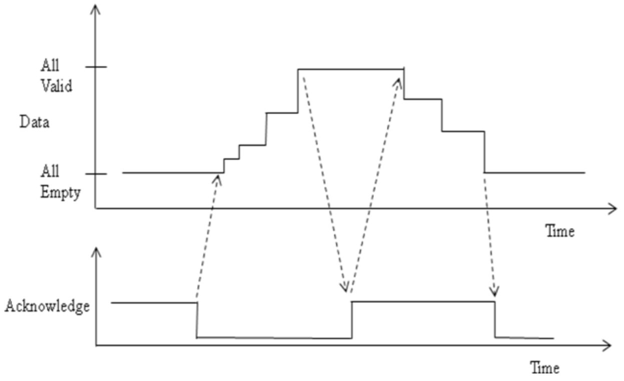

Fig.4 Illustration of the handshaking on a 4-phase dual-rail

Since the input is a continuous stream of valid data, an empty phase is inserted to distinguish between two consecutive valid data. This empty phase is generated using self-reset logic. The combinational blocks between the pipeline registers are reset when the data is latched in the register next to it. Since the acknowledge signal of the register immediately after the combinational block denotes the latching of data, the combinational block is reset using it.

3.2 The pseudo 4 phase protocol

The pseudo 4-phase protocol involves the 4 events as shown in Fig.5.

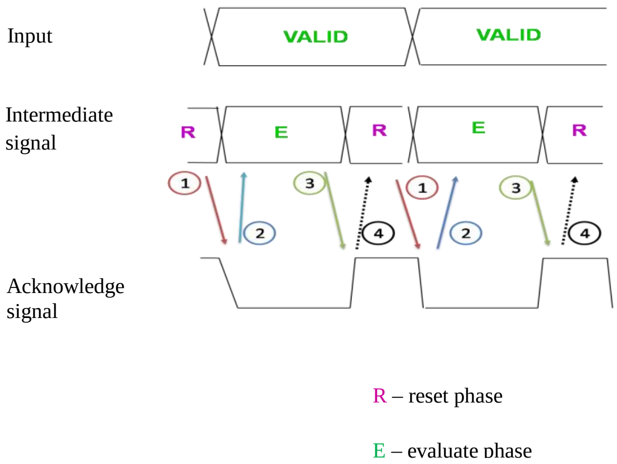

Fig.5 The pseudo 4 phase protocol

1. The reset phase pulls down the acknowledge
2. This lifts the reset and the combinational block is ready to process the next data(evaluate phase)
3. At the end of evaluation phase, a valid data is generated. As soon as the next register is ready to accept the data, the valid data is latched and acknowledge signal is generated.
4. The acknowledge signal resets the combinational block (reset phase) and so empty state is introduced.

To get back the valid data as a continuous stream (remove empty states), a latch with acknowledge signal generated using only the completion detector as a clock signal can be used.

3.3 Multiple Reset

In 4-phase dual-rail systems, the empty state has to be maintained for the period of the maximum combinational path delay. Since reset phase is equal to the evaluate phase in case of the usual 4-phase implementations, it increases with logic depth. This paper proposes a novel idea to overcome this problem. Each of the gates in the combinational block between two registers is reset simultaneously. So the reset phase is effectively the delay of the last gate in the combinational block and the delay of the completion detection circuit to detect the empty state. So the reset phase does not vary with the depth of the combinational block as illustrated in Fig. 6.

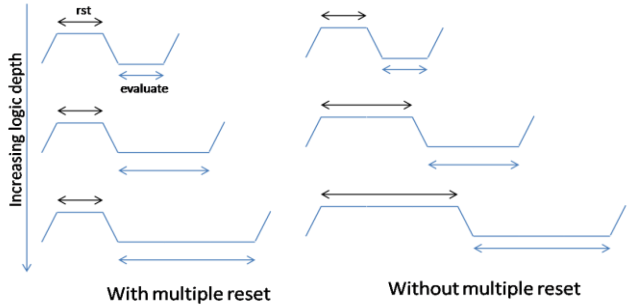

Fig.6 Model waveform of acknowledge showing the effect of Multiple-reset on reset and evaluate phase

4. APPLICATION: AN 8-BIT PARALLEL ADDER

We use the Carry Look Ahead structure proposed by [8]. The global organization is shown in Fig.7, where only the combinational block that performs the addition is depicted. This is a Carry Propagate Adder (CPA). This block by itself is essentially asynchronous. In order to make it work in a synchronous environment, an input register and an output register are added. The combinational block is composed of three stages: a propagate/generate block PG generator, followed by a parallel carry generator, and a sum generator. The maximum clock frequency depends on the cycle time of the gates, while controlling the maximum difference in arrival times of signals at any given stage**.**

The first stage forms the propagate term p and the generate term g, according to the following equations:

$$gi = ai \cdot bi, \quad i=1,2,\ldots n \qquad (1)$$

$$pi = ai \wedge bi,\ \ i=1,2,\ldots n \qquad (2)$$

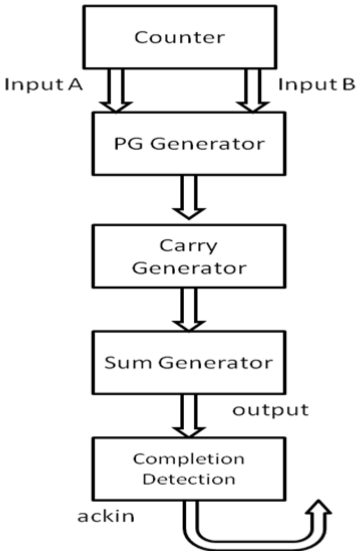

Fig.7 Carry propagate adder

The basic cell used in the PG generator is shown in Fig. 8. The resulting p and g bits are the inputs to the next stage, the parallel carry generator. Carries are computed from the recursive equations

$$ci = Gi \text{ for } i=1,2..n \qquad (3)$$

where

$$(Gi.Pi) = (gi, pi), \qquad \text{for } i=1 \qquad (4)$$

$$(Gi,Pi) = (gi+pi*Gi\text{-}1, pi*Pi\text{-}1) \text{ for } i=2, 3\ldots n \qquad (5)$$

Sums are computed from

$$si = pi \quad \text{for } i=1 \qquad (6)$$

$$s_i = p_i \wedge c_{i-1} \quad \text{for } i=2,3\ldots n \qquad (7)$$

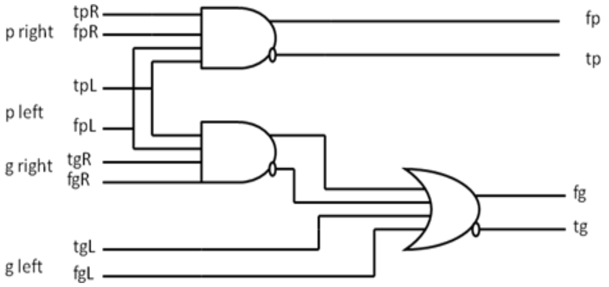

Fig.8 PG –cg cell

5. RESULTS

The 8-bit Carry Look Ahead adder was implemented in 0.18µm technology. The netlist file was generated using Leonardo Spectrum from the design file written in Verilog HDL. The layout design was (floor planning, place and route) carried out in Mentor Graphics IC Station tool. The parasitic extraction was done using Calibre and the post layout simulation was completed using Eldo SPICE. The number of pipeline registers introduced is shown in the Fig. 9, the number of registers was varied in order to investigate the variation of reset phase with logic depth. The logic depth is the number of stages of PG-cg cell between two registers. The results of variations of reset phase with increasing logic depth are provided in Table 1. Three cases of implementations are considered for verification for above mentioned proposal. The on time of the acknowledge signal generated gives the reset phase of the combinational block between the register. The period of acknowledge is the sum of evaluate and reset phase.

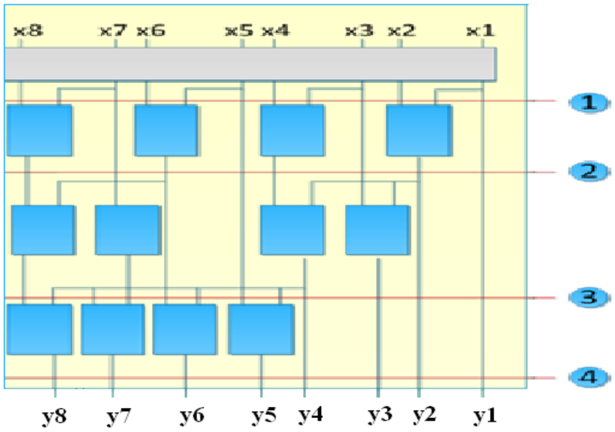

Fig.9 Carry generator block

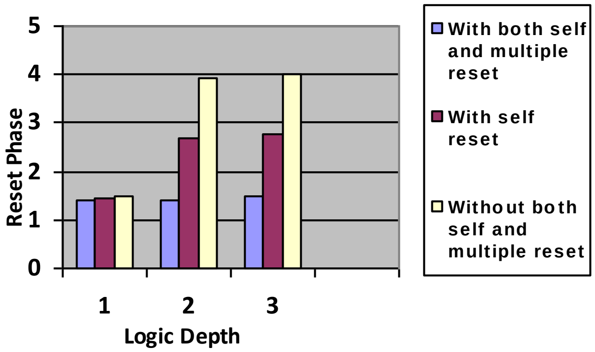

Fig. 10 Variation of reset phase with logic depth for various schemes

As seen from the graph in Fig.10 the reset phase of the combinational block in which both the self-reset and multiple-reset were used, is fairly constant irrespective of logic depth. Whereas, the blocks with no multiple-reset, the reset phase increases linearly with logic depth and also is equal to the maximum combinational path delay (evaluation phase). The reset period of the design with self-reset logic (without multiple-reset) decreased by 6%, 3%, 2% compared to the normal 4-phase protocol design for logic depth of 1 ,2 and 3 respectively. The reset period of the design with self-reset logic and multiple-reset decreased by 6%, 47.63% and 63.25% compared to the conventional 4-phase protocol design for logic depth of 1, 2 and 3 respectively.

Table 1 Comparison of period and reset phase for various schemes and logic depth

Logic Depth	With both Self and Multiple Reset		With Self-Reset (no multiple-reset)		Without Self and Multiple Reset (4-phase protocol)	
	Period (ns)	On Time (ns)	Period (ns)	On Time (ns)	Period (ns)	On Time (ns)
1	2.86	1.41(94%)	2.86	1.41(94%)	3.0	1.50(100%)
2	4.07	1.44(52.37%)	5.34	2.67(97%)	5.5	2.75(100%)
3	5.31	1.47(36.75%)	7.84	3.92(98%)	8.0	4.00(100%)

This shows that the reduction in reset phase varies inversely with logic depth. The layout of the implemented Carry Propagate Adder is shown in the Fig. 11. The waveforms obtained after post layout simulation using Eldo SPICE is shown in the Fig. 12.

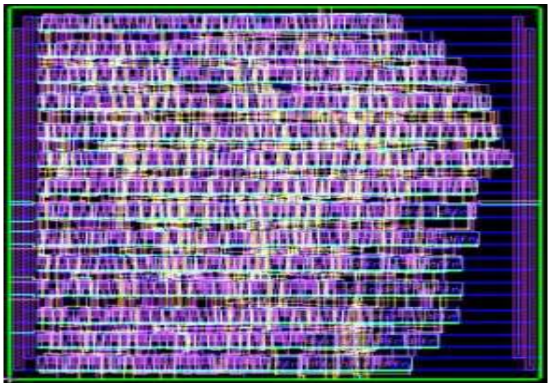

Fig.11 Layout of 8-bit carry propagate adder

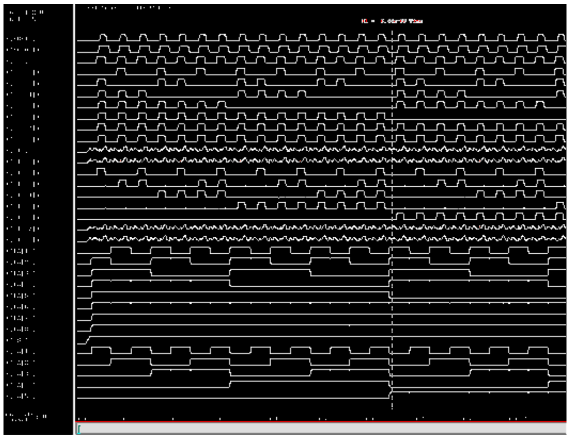

Fig.12 Post layout simulation waveforms

6. CONCLUSION

This paper presents a novel pseudo 4-phase protocol which involves self- reset logic for dual-rail applications. This protocol eliminates return-to-zero transitions in the global data lines and is simpler to build functional blocks using 4-phase dual-rail. The self-reset logic is used to reset the blocks whenever it has completed its evaluation and data is latched. The reset signals were generated using the completion detection circuit. Also multiple reset signals to all the gates of the combinational block between the two registers reduce the reset phase. The implementation results of 8 bit CPA adder in 0.18μm technology shows that the reset phase can be reduced up to 63.25% and 47.63% for a logic depth of 3 & 2 compared to the blocks without self and multiple-reset. This protocol with self-reset and multiple-reset concept increases the throughput rate of the system as it keeps the reset phase constant for increasing logic depth.

7. REFERENCES

[1] M. B. Josephs, S. M. Nowick, and C. H. K. van Berkel, “Modeling and design of asynchronous circuits,” Proc. IEEE, vol. 87, no. 2, pp. 234–242, Feb. 1999.

[2] C. L. Seitz, “System timing,” in: Introduction to VLSI Systems, C. A. Mead and L. A. Conway, Eds. Reading, MA: Addison-Wesley, 1980, ch. 7.

[3] Amitava Mitra William F. McLaughlin Steven M.Nowick Efficient Asynchronous Protocol Converters for Two-Phase Delay-Insensitive Global Communication IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'07) 0-7695-2771-X/07

[4] W. Hwang et al, “Implementation of a Self-Resetting CMOS 64-Bit Parallel Adder with Enhanced Testability,” IEEE Journal of Solid-State Circuits, Vol. 34, No.8, pp. 1108-1117, August 1999

[5] A. E. Dooply and K. Y. Yun, “Optimal Clocking and Enhanced Testability for High-Performance Self-Resetting Domino Pipelines,” Proceedings, 20th Conf. on Advanced Research in VLSI, pp. 200-214, 1999.

[6] Abdel Ejnioui and Abdelhalim Alsharqawi, “Pipeline Design Based on Self Resetting State Logic”, Proceedings of IEEE Computer Society Annual Symposium on VLSI Emerging Trends in VLSI Systems Design (ISVLSI’04), 2004 IEEE

[7] J. Sparsø and S. Furber (eds), “Principles of asynchronous circuit design - A systems perspective”, Kluwer Academic Publishers, 2001 (ISBN 0-7923-7613-7)

[8] W. Liu, T. Gray, D. Fan, W. J. Farlow, T. A. Hughes, and R. K. Cavin,“A250-MHzwave pipelined adder in 2-_mCMOS,” IEEE J. Solid-State Circuits, vol. 29, no. 9, pp. 1117–1128, Sep. 1994.