ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 5, Issue 7, July 2016

Design and Simulation of High speed 32 bit PASTA Architecture

J.Ramesh Naidu*,K.V.Ganesh ,V.Malleswara rao*****

*PG Student,Dept of ECE,VITAM College,Visakhapatnam ,Andhra Pradesh ,India

**Asst.Professor,Dept of ECE,VITAM College,Visakhapatnam ,Andhra Pradesh ,India

***Professor ,Dept of ECE ,GITAM University, Visakhapatnam, Andhra Pradesh, India

Abstract: **This describes a parallel adder. It is related to a repetitive formulation for doing multi bit binary summation. This operation is easier for the bits which does not require carry generation. Thus, the design achieves drastic speed over alternate operand conditions without any special speedup block. A complete execution is generated along with a detection unit. This accomplishes common and does not have any limitations of high fan outs. A high fan-in gate is required but this is not necessary for asynchronous logic and is connected by the transistors in parallel. Simulations have been performed using Xilinx 12.1 for synthesis and Model sim XE is used for simulation.**

Keywords: **Parallel adder(PADD); Fan out; Xilinx ;Model sim.**

1 INTRODUCTION

Processor important operation is Binary Summation. Adder circuits have been designed for synchronous blocks even though there is strength of interest in clock less/asynchronous blocks. Asynchronous designs[1] do not acquire any of time. Therefore, they hold powerful and potential for logic design as they are risk free from different problems of clocked blocks. The logic flow in asynchronous circuits is restrained by a request/ack, handshaking protocol to institute a pipeline in the absence of clocks. Observable handshaking designs for tiny elements, such as single bit adders, are expensive. Therefore, it is inheritantly and efficiently managed using double-rail carry propagation in adders. A valid double-rail carry output also generates acknowledgment from an adder block. Thus, asynchronous adders[1] are either based on full double-rail encoding of all signals or pipelined operation using single-rail data encoding and double-rail carry representation for acknowledgments[1]. While these constants add strength to circuit blocks, they introduce speed benefits of asynchronous adders[1]. Therefore, a more healthy alternative resemble is good consideration that can address these problems. This presents an asynchronous parallel adder using the algorithm[2]. The design of Parallel adder is simple and uses half-adders (HAs) along with mux's requiring minimum interconnections. Thus, it is suitable for Very Large Scale Integration execution. This design works with independent Carry chains. The execution in this paper is moderate it has a feedback from xor gate to generate a cyclic asynchronous adder. Cyclic circuits are more efficient than there acyclic blocks. Inputs are applied before the output get strengthen is known as wave pipelining[2]. It manages automatic pipelining of the generated carry inputs separated by propagation and inertial delays of the gates in the circuit path. Single-rail pipelined blocks are different from double-rail blocks.

2 BACKGROUND

There are numerous blocks of binary adders and we concentrate on asynchronous adders. The Self-Timed designs are nothing but industry standard designs. This type of adders are runs faster for data provided dynamically and early sensing can avoid worst case delay in synchronous circuits[2]. They are classified as.

2.1 Pipelined Adders Using Single-Rail Data Encoding

The asynchronous Request/Acknowledge handshake[2] can be used to start the adder block as well as to establish the flow of carry generation signals. In most of the cases, a double-rail carry convention is used for internal bitwise flow of carry outputs. These double-rail signals can represent more than two logic values (invalid, 0, 1), and therefore can be used to propagate bit-level acknowledgment when a single-bit operation is completed. When all ack bits are high complete detection unit will sense. The carry-completion sensing adder blocks is an example of pipeline adders, which uses full adder (FA) functional designs, adapted for double-rail carry.[2] A non-financially completion adder, It uses so-called different logic and early completion to select the number of delay lines for proper completion of response. However, the differed logic implementation is expensive due to high fan-in requirements[2].

2.2 Delay Insensitive Adders Using Dual-Rail Encoding

Delay insensitive (DI) adders are asynchronous adders that assign bundled constraints or DI operations. They can operate correctly in presence of bounded but unknown gate and net delays. There are many variants of DI adders, such as ripple carry adder and carry look-ahead adder[3]. These adders use double-rail encoding and are assumed to increase area. Though double-rail encoding doubles the net complexity, they can still be used to generate designs nearly as efficient as that of the single-rail variants using dynamic logic or nMOS designs[3]. DIRCA uses 40 transistors whereas RCA uses only 28 transistors. Similar to CLA, the DICLA defines carry propagation and kill equations in terms of double-rail encoding. They do not connect carry signals as chain but in a hierarchical manner. Thus, they can strongly perform more when there is long carry chain in a tree[3]. A further minimization is provided from the observation that double rail encoding logic can benefit from settling of either the 0 or 1 path. Double-rail logic does not wait for the both the paths to be executed. So it should speed up the CLA to send carry kill signals to any stage in the tree. This is developed and referred as DICLA with speedup circuitry (DICLASP).

3 DESIGN OF PASTA

In this section, theory and architecture of Parallel adder is presented. The adder first accepts two inputs and performs two half additions[4]. Subsequently, it starts iterates using previously generated carry and sums to perform half-additions recursively until all carry bits are consumed and settled at zero value.

ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 5, Issue 7, July 2016

Architecture of PASTA

The general architecture of the adder is shown in Fig. 1. The sel input of muxs are used as initially it selects the operands and when sel=1 used for carry paths. The feedback path from the HAs enables the multiple iterations to continue until the completion when all carry signals will assume zero values.[4]

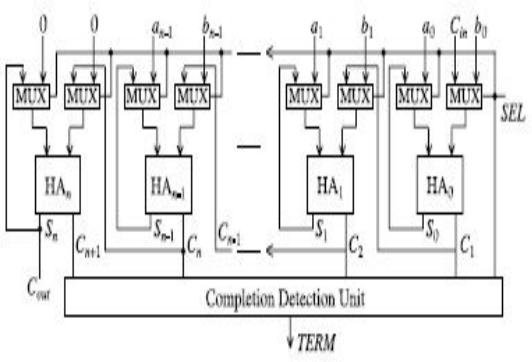


Fig. 1. General Block Diagram of PASTA

3.1 State Diagrams

In Fig. 2.2 state diagrams are given for the initial phase and the iterative phase of the proposed design. Each state is represented by (*Ci*+1, *Si*) pair where *Ci*+1, *Si* is carryout and sum values, respectively, from the *i*th bit adder block[4]. During the initial phase, the circuit works as a combinational half adder operating in normal mode. Due to the usage of half adders instead of full adders, state cannot appear.

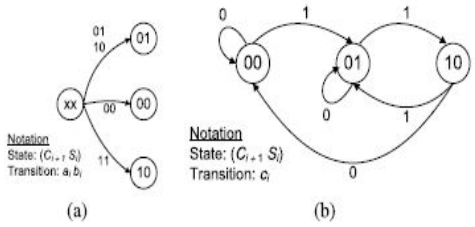


Fig. 2. State diagrams for PASTA. (a) Initial phase. (b) Iterative phase.

During the iterative phase (SEL = 1), the feedback path through mux block is activated. Till Completion of the recursion the carry transitions (*Ci*) are allowed. From the definition of normal mode designs, the present design cannot be considered as a normal mode circuit as the input–outputs will go through several transitions before producing the final output. Several transitions will take place, as shown in the state diagram[4]. This is analogous to cyclic sequential circuits where gate delays are utilized to separate individual state.

3.2 Recursive Formula for Binary Addition

Let S_i^j and C^j_{i+1} denote the sum and carry, respectively, for *i*th bit at the *j* th iteration. The initial condition (*j* = 0) for addition is formulated as follows:[4]

$$S_i^0 = a_i \wedge b_i \qquad (1)$$

$$C^0_{i+1} = a_i \,.\, b_i. \qquad (2)$$

The *j* th iteration for the recursive addition is formulated by

$$S_i^j = S_i^{j-1} \wedge C_i^{j-1}, \quad 0 <= I < n. \qquad (3)$$

$$C_{i+1}{}^j = S_i^{j-1} \wedge C_i^{j-1}, \quad 0 <= I <= n. \qquad (4)$$

The recursion is terminated at *k*th iteration when the following condition is met:

$$C^k_n + C^k_{n-1} + \ldots + C^k_1 = 0, \quad 0 <= k <= n. \qquad (5)$$

ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 5, Issue 7, July 2016

4 IMPLEMENTATION

The PASTA architecture can be implemented using Xilinx12.1software for synthesis and Modelsim XE is used for simulation.[4]The completion detection following is negated to obtain an active high completion signal (TERM). This requires a large fan-in *n*-input NOR gate. Therefore, an alternative more practical pseudo-nMOS ratio-ed design is used. The resulting design is shown in Figure Using the pseudo-nMOS design, the completion unit avoids the high fan-in problem as all the connections are parallel[4]. The pMOS transistor connected to Vdd of this ratio-ed design acts as a load register, resulting in static current drain when some of the nMOS transistors are on simultaneously[5]. In addition to the *Ci* s, the negative of SEL signal is also included for the TERM signal to ensure that the completion cannot be accidentally turned on during the initial selection phase of the actual inputs. It also prevents the pMOS pull up transistor from being always on[5]. Hence, static current will only be flowing for the duration of the actual computation. VLSI layout has also been performed Fig.(d) for a standard cell environment using two metal layers. The layout occupies 270 λ × 130 λ for 2-bit resulting in 1.123 Mλ2 area for 32-bit.

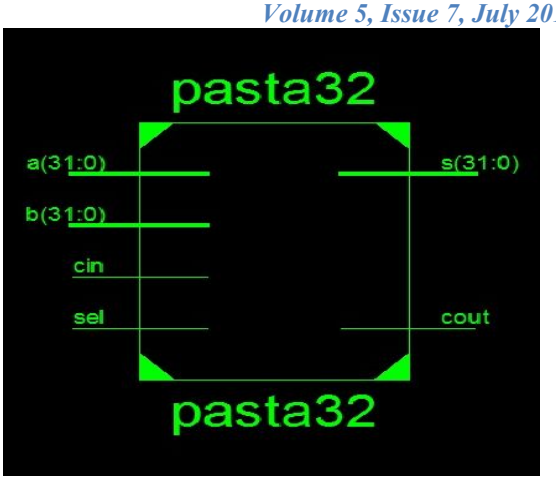
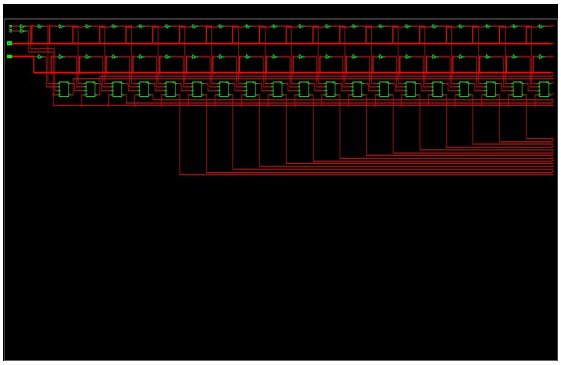
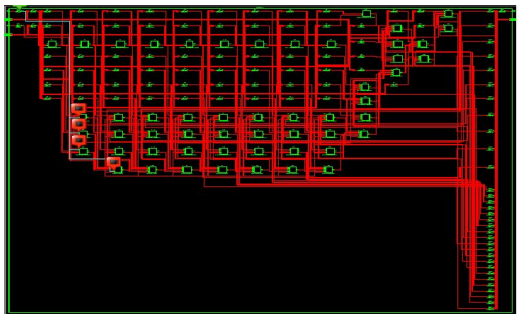


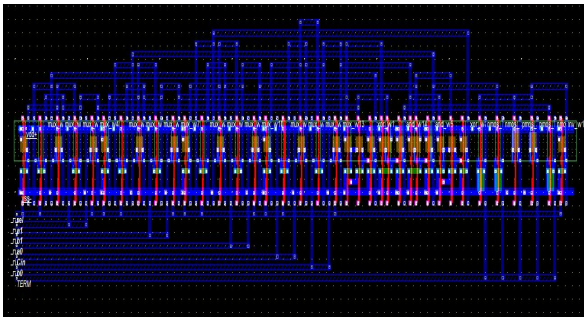
Fig (a) pin diagram



Fig(b)Technology schematic

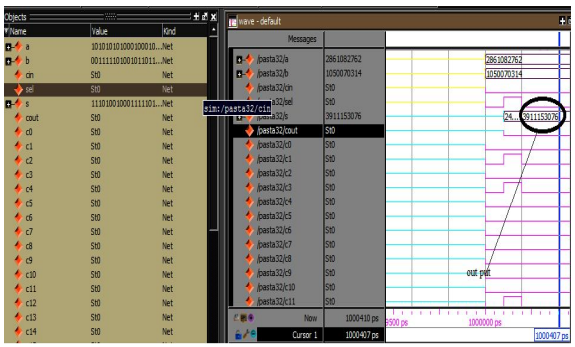


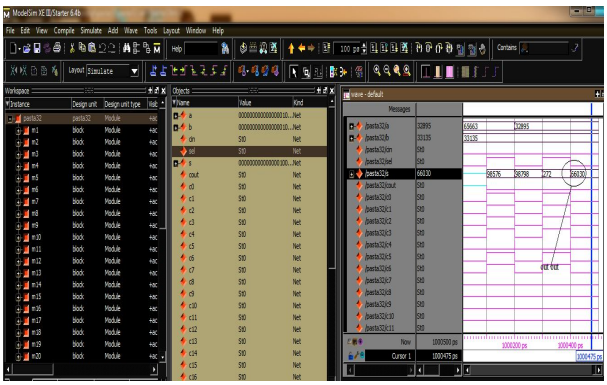
Fig(c)RTL schematic



Fig(d) Layout

5 SIMULATION RESULTS





CONCLUSION

This paper presents an expense execution of PASTA. The design acquires a very simple 32-bit adder that is area efficient and fewer interconnections like Ripple Carry Adder. The circuit works in a way such that it achieves logarithmic average time speed over randomly generated values. The Detection unit of the Parallel adder is also consumes less area and delay as validated. Synthesis can be performed using Xilinx 12.1 and Simulation results are also verified using Model sim XE

REFERNCES

[1] D. Geer, “Is it time for clockless chips? [Asynchronous processor chips],” *IEEE Comput.*, vol. 38, no. 3, pp. 18–19, Mar. 2005.

[2] J. Sparsø and S. Furber, *Principles of Asynchronous Circuit Design*. Boston, MA, USA: Kluwer Academic, 2001.

[3] P. Choudhury, S. Sahoo, and M. Chakraborty, “Implementation of basicarithmetic operations using cellular automaton,” in *Proc. ICIT*, 2008,pp. 79–80.

[4] M. Z. Rahman and L. Kleeman, “A delay matched approach for the design of asynchronous sequential circuits,” Dept. Comput. Syst.Technol., Univ. Malaya, Kuala Lumpur, Malaysia, Tech. Rep. 05042013,2013.

[5] M. D. Riedel, “Cyclic combinational circuits,” Ph.D. dissertation, Dept.Comput. Sci., California Inst. Technol., Pasadena, CA, USA, May 2004.