Implementation of Parallel Prefix Adders Using FPGA’S

1Avneet Kaur and 2Chanpreet Kaur Toor

1Student, Electronics and Communication Engineering, Chandigarh Engineering College, Landran, Avneet Kaur Ludhiana, Punjab.

Address For Correspondence:

ARTICLE INFO
Article history:
Received 18 February 2017
Accepted 15 May 2017
Available online 18 May 2017

Keywords:

ABSTRACT

Objectives: This paper examines three variations of carry-tree adders (spanning tree adder, sparse Kogge-Stone & Kogge-Stone adder) and presents a comparison with Carry Skip Adder (CSA) & simple Ripple Carry Adder (RCA). Methods/Statistical Analysis: In VLSI designs the best performance can be obtained from Parallel-prefix adders (PPA) which are also termed as carry tree adders. Due to constraints on routing overhead & logic block configurations the advantage in performance in FPGA executions doesn’t translate directly. Xilinx Spartan 3E FPGA is used to implement these designs which has varied bit-widths & the -performance logic analyzer is used for delay dimensions. Findings: Better delay performance is observed by the RCA designs of 128 bits due to the use of fast carry-chain. As in the case of carry-tree adders having bit widths of 256 bits there is a speed advantage over the RCA. RCA is the most commonly used adder implemented using full and half adders. As RCA is a serial adder which can perform any no. of additions but due to carry propagation there is a problem of propagation delay from stage to stage. PPA are used to overcome delay problem as the carry is pre-computed. Various PPAs are Spanning tree, SKS adder, KS adder and Brent kung adders. Improvements: By using delay and power controls, design and comparison between these adders performed. Xilinx ISE software is used for synthesis process & simulation of these adders and results are presented in this paper.

INTRODUCTION

In the binary addition unit, the main component in the majority digital path classes together with digital signal processors as well as micro chip find out passageway units. Generally, severe examination carries on being beneath fire at rising the capability holdup time of the adder (Hoe, D.H.K., et al., 2011). Reconfigurable logics such as Field Programmable Gate Arrays are most popular from the past few years and are latest in technology. As significance, it provides us with the benefits of improved rate and power for DSP and microprocessor based resolutions for abundant reasonable styles including mobile DSP and telecommunications applications as well as a deep degradation in progress time as well as price on to Application Specific computer circuit designs. The skill benefit is mainly key with the increasing value of mobile as well as portable physical science, which causes austere use of DSP functions (Yeyerla, S.K. and B. Rajendra Naik, 2014).

Though, because of the structure of the configurable logic as well as steering income in FPGA, parallel-prefix adders can have an improved performance than Very Large Scale Integration implementations. Field Programmable Gate Arrays which are up to date use a fast-carry chain that enhances the carry passageway for the Ripple Carry Adder. From this paper, problems appeared in coming with as well as designing tree-based
adders on Field Programmable Gate Arrays are labeled. Assistant degree in luxurious testing criteria for contrasting the feat of those adders is described (Roy, S., et al., 2014). Numerous tree-based adder designs are compulsory and featured on a Field Programmable Gate Arrays as well as estimated with the Ripple Carry Adder as well as also the Carry Skip Adder.

**Parallel Prefix Adders:**
The PPA is sort of a Carry Look Ahead Adder. The prefix adders will be designed in a number of different ways that endured the innumerable wants. There is a bent to use tree structure sort for expanding the quickness of process. Parallel prefix adders are faster adders and these are rapider adders. They are used for major presentation arithmetic structures in trades and assemblies (Zode, P.P. and R.B. Deshmukh, 2014; Escobar, K.A. and R.P. Ribas, 2013). The parallel prefix addition is completed in three steps.

A. **Pre-Processing Stage:**
In this stage we are inclined to work out, produce and broadcast signals. Carry input of every adder is generated (Nehru, K., A. Shanmugam and S. Vadivel, 2012). A and B are the inputs. These signals are given by the equation 1& 2.

\[ P_i = A_i \text{xor} \ B_i \]  
\[ G_i = A_i \cdot B_i \]  

B. **Carry Generation Network:**
In this step there is similarity to work out carries such as every bit. Achievement is finished in parallel type (Martinez, C.D., et al., 2012). When the working out of carries is in parallel, they're divided into smaller items. Carry operator contain 2 AND gates, one OR gate. It practices, propagate and generate as midway signals that are given by the equations 3& 4.

\[ R_{i,k} = P_{i,k} \cdot P_{i-1,k} \]  
\[ G_{i,k} = G_{i,j} + (G_{j-1,k} \cdot P_{i,j}) \]  

C. **Post Process Stage:**
This is the final period to work out the neglected of input bits. This is similar for all adders. The add bit equation given

\[ S_i = P_i \text{xor} \ C_i \]  
\[ C_{i+1} = (P_i \cdot C_0) + G_i \]  

**Proposed Methodology:**
In the proposed approach a high speed parallel prefix adder is proposed. In the proposed methodology, a high speed kogge stone adder is realized using high speed adder logic. The overall performance of the system is aimed to be improved in conditions of both area and timing complexity. In the proposed method the carry select adder is divided into further four blocks. In the first unit simple half adder logic is implemented. This
block is the mutual block for both the carry ‘1’ and ‘0’ mathematical calculation. The methodology is shown in figure 2 with different blocks.

Fig. 2(a): Half Adder Unit

Fig. 2(b): Carry and Sum Generation Unit for Cin

Fig. 2(c): Carry and Sum Generation Unit for Cin_bar

Fig. 2(d): Multiplexer Unit
Fig. 2(e): Proposed Carry Select Adder

In figure 2(e), the proposed CSLA is shown. The division of adder architecture into 4 sub blocks is shown in figure 2(a)-2(d). In the first block the simple half adder is used which figures the sum and carry value which is shown in figure 2(a). $S_i$ is used for sum calculation and $C_i$ is used for carry output. The Half adder sum and carry output equations are

$$S_i = a \text{ xor } b$$
$$C_i = a \text{ and } b$$

In the second block Carry and Sum generation is realized which after taking the output from the first unit generates the sum and carry values ($C_0, S_0$ used for carry and sum respectively) for input carry $C_{in}$. The whole architecture behaves like a full adder realized using two half adder blocks. The output of the sum and carry are produced using

$$S_0 = S_i \text{ xor } C_i$$
$$C_0 = (S_i \text{ and } C_{in}) \text{ or } C_i$$

The third block also behaves like a full adder. It shares the common half adder resources from the first block and after that generates the carry and the sum ($C_1, S_1$ for carry and sum of this block respectively) for carry input as $C_{in}$ bar. Finally in the fourth block, the mux architecture is implemented and is shown in figure 2(d). It selects one of the outputs from the second and third block. The final sum is generated by using the equation

$$S = (S_0 \text{ and not } C_{in}) \text{ or } (S_1 \text{ and } C_{in})$$

**RESULTS AND DISCUSSIONS**

The proposed technique is executed using the Xilinx Spartan 3 FPGA and Verilog language is used for the implementation of the design. The technique is associated with the basic RCA approach and the results show an enhancement in both the area consumed by the device and the maximum combinational path delay. In Table 1 the comparison of both the approaches is shown on the basis of various performance parameters like number of slices, number of look up tables, delay and maximum operating frequency. Number of slices, number of 4 input LUTs are a amount of the number of resources used by the FPGA which comes out to be less in case of the proposed approach.

**Table 1: Comparison Table**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Basic Approach</th>
<th>Proposed Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Slices</td>
<td>52</td>
<td>46</td>
</tr>
<tr>
<td>Number of 4 Input LUTs</td>
<td>92</td>
<td>81</td>
</tr>
<tr>
<td>Number of Bonded IOs</td>
<td>101</td>
<td>101</td>
</tr>
<tr>
<td>Delay (ns)</td>
<td>33.166</td>
<td>21.913</td>
</tr>
<tr>
<td>Max. Frequency (MHz)</td>
<td>30.15</td>
<td>45.63</td>
</tr>
</tbody>
</table>
While delay is the measure of the speed of design which also comes out to be less as compared to the basic approach. Figure 3 shows the top level module and RTL view of the methodology which is proposed and figure 4 shows the simulation waveform. It is lucid from the comparison chart that the proposed methodology utilizes less area in comparison with the basic ripple carry adder. The performance is faster and decreases the maximum combinatorial path delay of the carry select adder.

**Fig. 3: RTL Schematic of the proposed Methodology**

**Fig. 4: Simulation Waveform of Proposed Methodology**
The graph shows the comparison outcomes in figure 5 and 6. The comparison of the area utilization by the device is shown in figure 5 and in figure 6 the proposed delay improvement is shown. Figures show the comparison of data shown in table 1. It is clear from the figure 5 that the proposed approach utilizes lesser area as compared to the basic approach. It is shown in terms of number of slices and number of 4 input LUTs. Figure 5 shows the comparison of combinational path delay which is the measure of the speed of the design the minimum delay, the maximum speed and it is clear from the graph that proposed approach has less delay and maximum operating frequency as compared to the basic approach.

Fig. 5: Comparison of Area Utilization

Fig. 6: Comparison of Delay

Conclusion:

Arithmetic unit is the most widely researched and used designs in the digital electronics circuits. Many researchers in recent past have proposed different approaches for the design of adders and other logical circuits and compared their merits and demerits with the existing approaches. Parallel prefix adder is designed in this approach by using optimized CSLA in terms of area and speed. In the proposed approach carry select adder is planned with the help of logical gates for reducing the area consumed by the adder. It is shown from the results that the proposed approach outperform the basic approach on the basis of both area and delay of the design.

REFERENCES


