DESIGN AND IMPLEMENTATION OF LOW POWER HIGH SPEED 32-BIT HCSA
YOGAYATA SHRIVASTAVA¹, TARUN VERMA², RITA JAIN³
Department of ET, LNCT, Bhopal, India
¹shri.yogayata@gmail.com, ²tarunindia@rediffmail.com, ³ritajain_bpl@gmail.com

ABSTRACT

In the design of carry select adder, the requirement of area, speed and power consumption is of prime importance. Power dissipation is one of the most important design objectives in integrated circuits, after speed. Carry select adder (CSLA) is one of the fast adder used to perform fast arithmetic operations as we select the carry beforehand and calculate the sum output for both the carry conditions i.e. for Clatent=1 or Clatent=0. The most fundamental arithmetic operations in any ALU is addition. It has been ranked the most extensively used operation among a set of real-time digital signal processing benchmarks from application-specific DSP to general purpose processors [1]. Efficient utilization of CSLA depends upon the gate level modification. In this paper, 32-bit linear CSLA has been modified in such a way that the pipeline architecture that we have developed consumes minimum power as well as provide a high speed. In this project, the entire adder architecture has been implemented using VHDL and simulated using Mentor Graphics tool suite. The architecture is then further evaluated in FPGA using Xilinx ISE Design Suite.

Keywords: Arithmetic logic unit (ALU), Configurable Logic Blocks (CLB), Carry Select Adder (CSLA), Field Programmable Gate Array (FPGA), High-speed carry select adder (HCSA), Look up-tables (LUT), Ripple carry adder (RCA).

[1] INTRODUCTION

Adders are of fundamental importance in a wide variety of digital systems. Various adders are available today but adding fast using low area and power is still challenging. In any data processors, data is processed using binary operations where addition is one of the most fundamental operations required to perform any operations. Adders are not only required for every arithmetic operation, but are also required to locate physical address in every memory fetch operations in CPUs. Thus for this purpose we have developed various adders and Carry Select Adder (CSLA) is one of them. Carry select adder is a best choice among all the available addresses especially in the case of carry delay. This is because this adder pre-calculates the expected outcomes. As we know that in a ripple carry adder, every full adder cell has to wait till the incoming carry will come and then an outgoing carry can be generated. This dependency can be eliminated by pre-calculating i.e. by taking both possible values of the carry input and evaluating the result for both possibilities in advance. As we come to know the actual value of the incoming carry, then correct result is easily selected by using a simple multiplexer stage. The implementation of this idea is called the linear carry select adder.

Addition usually influences widely the whole performance of digital systems and a complicated arithmetic function. In electronic applications adders are essentially required. Applications where these are used are multipliers, DSP to execute various algorithms like FFT, FIR and IIR. As we know that in microprocessors millions of instructions are performed per second. So, speed of operation is the most important constraint to be considered while designing multipliers. Due to the demand of device portability increases it requires miniaturization of the processors and speed must be high. Devices like Mobile, Laptops etc. require more battery backup thus power consumption should be very low. As we know that the number of applications is increasing day-by-day and so the battery consumption, thus the need to conserve the power is one of the major challenges today. So, a VLSI designer has to optimize these three parameters in a design. These constraints are very difficult to achieve so depending on demand or application some compromise between constraints has to be made. Minimizing Area and delay has always been considered important, but Reducing power consumption has been gaining prominence recently with the increasing level of device integration and the Growth in complexity of micro-electronic circuits, reduction of Power dissipation has come to fore as a primary design goal as power efficiency has always been desirable in electronic Circuits.

Addition is the core of many other useful operations such as, subtraction, multiplication, address calculation and etc. It is also the speed limiting and more power consuming element as well. The design of faster, smaller and more efficient adder architecture has been aim and goal for many research efforts and has resulted in a large number of adder architectures. The power consumption and propagation delay are two most important properties of the adder circuit architectures which basically are against each other. That is known, lowering the power causes longer propagation delay and vice versa, hence, most architectures referring to one of those important properties. Nevertheless, in some cases they both may compromise to achieve to low energy consumption. All architectures provide different insight and therefore require different implementation. Here the architecture of
CSLA that we have implemented will provide significant reduction in power consumption as well as improvement in speed.

This brief is structured as follows. Section II and III deals with the Boolean logic full adder function used in RCA and the delay of n-bit carry select adder respectively. Section IV presents the detailed structure and the function of the proposed CSLA. Section V discussed about the SQR·T CSLA and modified SQR·T CSLA has been chosen for comparison with the proposed design as it has a more balanced delay, and requires lower power and area [1], [2]. The simulation result obtain and comparison with the previous work will be discussed in the section VI, VII respectively. Finally, the work is concluded in Section VIII.

II. Boolean Logic Function for Full Adder and RCA

A full adder Boolean logic function is based on three inputs, (Ai, Bi, Ci-1) and provides two outputs Si and Ci.

Equation (1.1) and (1.2) are sum and carry outputs with respect to their input.

\[ Si = Ai \oplus Bi \oplus Ci-1 \]  
\[ Ci = Ai Bi + Bi Ci-1 + Ai Ci-1 \]  

However it is most common and practical to use denoted characters Pi (Carry Propagate) and Gi (Carry Generate) and rewrite (1.1) and (1.2) by replacing (1.3) and (1.4).

\[ Pi = Ai + Bi \]  
\[ Gi = Ai Bi \]  

So sum and carry outputs is given by:

\[ Si = Pi \oplus Ci-1 \]  
\[ Ci = Gi + Pi Ci-1 \]  

There are different solutions to implement n-bits full adder. First, architecture must be defined based on speed or power consumption.

2. Ripple Carry Adder

The simplest addition architecture is based on a linear array of a full adder cell as it is depicted in figure (1.1). The architecture of RCA has been subjected to be the smallest and the lowest power consuming. However according to the experimental results obtained in this project, they show the average activity overhead (glitch) that is about 50% [2]. The critical delay path or worst case delay in n-bit RCA is given by:

\[ t_p = (n - 1) \times t_{carry} \]  

Half Adders can be used to add two one bit binary numbers. It is also possible to create a logical circuit using multiple full adders to add N-bit binary numbers. Each full adder inputs a Cin, which is the Cout of the previous adder. This kind of adder is a Ripple Carry Adder, since each carry bit "ripples" to the next full adder. The first (and only the first) full adder may be replaced by a half adder.

The block diagram of 4-bit Ripple Carry Adder is shown here below:

The layout of ripple carry adder is simple, which allows for fast design time; however, the ripple carry adder is relatively slow, since each full adder must wait for the carry bit to be calculated from the previous full adder. The gate delay can easily be calculated by inspection of the full adder circuit. Each full adder requires three levels of logic.

III. Carry Select Adder

Carry Select Adder is one of the fastest adders used in many data-processing processors to perform fast arithmetic functions. The CSLA is used in many computational systems to alleviate the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum [1].
Carry select Adder is a better choice especially in the case of Carry delay. As in a ripple-carry adder, every full adder cell has to wait for the incoming carry before an outgoing carry can be generated. This dependency can be eliminated by pre-calculating i.e. by taking both possible values of the carry input and evaluating the result for both possibilities in advance. Once the real value of the incoming carry is known, the correct result is easily selected with a simple multiplexer stage. Now a days, design of low power, area efficient and high speed data path logic systems are the most substantial areas in the research of VLSI design.

On the basis of requirements such as area, delay and power consumption some of the complex adders are Ripple carry adder, Carry look-ahead adder and Carry select adder. Ripple carry adder (RCA) shows the compact design but their computation time is longer. Time critical applications make use of Carry Look-Ahead Adder (CLA) to derive fast results but it leads to increase in area. But the carry select adder provides a compromise between the small areas but longer delay of RCA and large area with small delay of Carry Look Ahead adder. The basic idea is to implement the pipelined structure of CSLA in such a way that the number of stages it have will not activated for the whole duration. Only that stage will be activated through which carry is propagating. The main advantage of this is we are successful in reduction of power consumption. The details about this will discuss in the section III.

Each of these sections is composed of two 4-bits ripple-carry adders. This is referred as linear expansion. The delay of n-bit carry select adder based on an m-bit CLA blocks can be given by the following equation when using constant carry number blocks.

\[ T = t_{\text{setup}} + m \cdot t_{\text{carry}} + \frac{(n/m)}{t_{\text{mux}}} + t_{\text{sum}} \]  

And by the following equation when using successively incremented carry number blocks respectively.

\[ T = t_{\text{setup}} + m \cdot t_{\text{carry}} + (2n)^{1/2} \cdot t_{\text{mux}} + t_{\text{sum}} \]

IV. PROPOSED HCSA ARCHITECTURE

In this paper we are implementing a linear CSLA structure with the use of RCA because RCA are the faster one but consume more power so for the power reduction we are implementing the structure in uniform bit size of 32-bit and add the all at ones without breaking them into 4, 8 and 16 bit. In this way the propagation delay will reduce and the proposed CSLA become faster than the other CSLA present today. In order to enhance the speed further we use the vertex-5 to written the VHDL code which is faster than the other traditional one and then we implement it using FPGA in XILINX 14.1 ISE Design suit. As we know that in FPGA all the sources are describe by using the CLB’s whose speed is quite high. The data arrival time of this adder is much less as we compared it with the previous adders. Therefore the adder which we have developed here is very fast than the others.

The other constraint over which we focussed our attention is the power consumption of the CSLA. Now the reduction of power requirement of the CSLA is also a major factor which we have relatively reduced to a considerable amount than the conventional CSLA required.
The basic idea is to implement the pipelined structure of CSLA in such a way that the number of stages it have will not activated for the whole duration of time while the only that stage is to be activated in which the computation is processing i.e. the only that pipelined stage kept activated through which carry is propagating. As the carry runs from one stage to the other than the previous pipelined stage will be deactivated and next stage is activated. Thus in this manner we are able to reduce the power consumption to a comparatively very low level than the carry select adders we are using today.

V. DELAY AND POWER EVALUATION

The delay of any basic building block of CSLA can be calculated as the number of gates is required for the computation of the input given. Here each gate is contributing a delay of one unit. Now the power evaluation can be calculated by simulating these blocks in XILINX 14.1 ISE design suite. As we know that to implement the CSLA the basic building block requirement is RCA, and it is made up of Full adder. So, we have to measure the delay and power of FA and RCA blocks. These basic blocks now combined into groups say group2, group3, group5 and group6.

The n-bit RCA is constructed by using the n number of full adders. Here the concept we can use that the first full adder can be replaced by half-adder because there is no carry-in for the first adder hence the delay of three gates can be compensated.

As from [1], the calculated delay they obtained is compared with the delay of this work which is greater and thus the latency, means the time required to change the output as soon as the inputs are changing is less in this work than the obtained by the [1].

Table I – Delay count of SQRT CSLA

<table>
<thead>
<tr>
<th>Group</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group2</td>
<td>11</td>
</tr>
<tr>
<td>Group3</td>
<td>13</td>
</tr>
<tr>
<td>Group4</td>
<td>16</td>
</tr>
<tr>
<td>Group5</td>
<td>19</td>
</tr>
<tr>
<td>Total</td>
<td>59</td>
</tr>
</tbody>
</table>

Table II – Delay count of Modified SQRT CSLA

<table>
<thead>
<tr>
<th>Group</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group2</td>
<td>13</td>
</tr>
<tr>
<td>Group3</td>
<td>16</td>
</tr>
<tr>
<td>Group4</td>
<td>19</td>
</tr>
<tr>
<td>Group5</td>
<td>22</td>
</tr>
<tr>
<td>Total</td>
<td>70</td>
</tr>
</tbody>
</table>

Table III – Delay count of Proposed CSLA

<table>
<thead>
<tr>
<th>Group</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group2</td>
<td>10</td>
</tr>
<tr>
<td>Group3</td>
<td>12</td>
</tr>
<tr>
<td>Group4</td>
<td>15</td>
</tr>
<tr>
<td>Group5</td>
<td>19</td>
</tr>
<tr>
<td>Total</td>
<td>56</td>
</tr>
</tbody>
</table>

Comparing Tables I, II and III, it is clear that the proposed linear CSLA reduces 14 gate delays than the regular SQRT CSLA, and 3 gate delays than the modified SQRT CSLA. To further evaluate the performance in terms of power, we have resorted to simulation and FPGA implementation.

VI. SIMULATION RESULT

The proposed design in this paper has been developed using VHDL and simulated using Mentor Graphics tool suite. The architecture is then further evaluated in FPGA using Xilinx 14.1 ISE Design Suite. To optimize the better result in terms of speed we have taken some measures in design suit.
In Xilinx the design goal and strategy is kept balanced and the optimization goal is set to speed so that the speed improve. The device utilization summary is given here as the number of slice flip-flops used is 96 from the available 1,536, number of input LUT’s used are 121 from the available 1,536, the number of occupied slices are 101 while the available is 768 and the number of slices containing only related logic is 100% utilized whereas the number of slices containing unrelated logic is 0%. Thus the program written is very specific which doesn’t contain any illogical like thing. The basic layouts, RTL schematic and final output obtain from the design suit are given below:

![Simple layout of 32-bit HCSA](image1)
![Technological view of 32-bit HCSA](image2)
![Final output of 32-bit HCSA](image3)

VII. COMPARISION
The delay and power obtained after simulating the HCSA in Xilinx 14.1 ISE design suite and implementing in FPGA we obtained the following results by which we can compare this work with the previous works. The obtained maximum delay of HCSA is 0.589ns which is much lesser than the regular CSLA as well as the modified CSLA. Thus the proposed CSLA is of very high speed than conventional one and therefore we named HCSA i.e. High-speed Carry Select Adder. The comparison between the delay and power is summarized in table below:

<table>
<thead>
<tr>
<th>Word Size</th>
<th>Adder</th>
<th>Delay (nS)</th>
<th>Power (µW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular CSLA</td>
<td>5.137</td>
<td>1127.3</td>
<td></td>
</tr>
<tr>
<td>Modified CSLA</td>
<td>5.482</td>
<td>969.90</td>
<td></td>
</tr>
<tr>
<td>HCSA</td>
<td>0.589</td>
<td>890.37</td>
<td></td>
</tr>
</tbody>
</table>

CONCLUSION
A simple approach is proposed in this paper to reduce the delay and power of Linear CSLA architecture. The idea to provide the power to a single pipelined stage through which carry is propagating and cut-off the power supply to all the other pipelined stages reduced the power consumption to a low level. This work offers the great advantage in the reduction of power and also the delay.

The compared results show that the HCSA is more efficient in terms of power requirement and speed than the modified SQRT CSLA and regular CSLA but the area has to be compromise.

The obtained HCSA architecture is therefore, High speed, low power, simple and efficient for VLSI hardware implementation.

ACKNOWLEDGEMENTS
The authors would like to thank Majhar-ul-Islam and Soheb Islam of the ET department, LNCT, Bhopal, India, for their contributions to this work.
REFERENCES