International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 06 | June 2020 www.irjet.net p-ISSN: 2395-0072

A LOW POWER CONSUMPTION AND HIGH-SPEED CARRY LOOK AHEAD ADDER BY USING CARRY MASKABLE ADDER FOR APPROXIMATE COMPUTING

[1]Muppidi Saritha Devi, [2]Kuppili Rubeeshwara Rao, [3]Gedala Ajay Kumar, [4]Chatti Venkata Rao, [5]Pathala Tejas

[1]Assistant Professor, [2]UG Student, [3]UG Student, [4]UG Student, [5]UG Student
[1,2,3,4,5]Department of Electronics and Communication Engineering
[1,2,3,4,5]Godavari Institute of Engineering and Technology, Rajahmundry, India

---

Abstract: In this paper, we proposed a low-power yet high speed carry look ahead adder by using carry maskable adder that also maintains a small design area. The proposed adder is based on the conventional carry look-ahead adder, and its configurability of accuracy is realized by masking the carry propagation at runtime. Compared with the traditional carry look-ahead adder, the proposed 16-bit adder reduced more power consumption, since we are using less number of LUT'S the path delay is less according to the accuracy configuration settings, respectively. Furthermore, compared with other previously studied adders, the experimental results demonstrate that the proposed adder achieved the first purpose of optimizing both power and speed simultaneously without reducing the accuracy.

Keywords: Carry look-ahead adder, accuracy-configurable adder.

I. INTRODUCTION

Applications that have recently develop (such as image recognition and synthesis, digital signal processing, which is computationally demanding, and wearable devices, which have needed battery power) have created challenges contingent to power consumption. Addition is a key fundamental arithmetic function for these applications. Most of these applications have an inherent tolerance for insignificant inaccuracies. By exploiting the inherent tolerance feature, approximate computing can be adopted for a trade off between accuracy and power. At present, this trade off plays a significant role in such application domains. As computation quality requirements of an application may vary significantly at runtime, it is preferable to design quality configurable systems that are able to trade off computation quality and computational effort according to application requirements. The previous proposals for configurability suffer the value of the rise in power or in delay. In order to profit such application, a low-power and high-speed adder for configurable approximation is strongly required. In this paper, we propose a configurable approximate adder, which consumes lesser power than does with a comparable delay and area. In addition, the delay observed with the proposed adder is much smaller than that of with a comparable power consumption. Our primary contribution is that, to achieve accuracy configurability the proposed adder achieved the access of power and delay simultaneously and with no bias toward either. We implemented the proposed adder, the conventional carry look-ahead adder (CLA), and the ripple carry adder (RCA) in Verilog HDL using a 45-nm library. Then we evaluated the power consumption, critical path delay, and design area for each of these implementations. Compared with the traditional CLA, the proposed adder reduced power consumption and critical path delay better than conventional adder, respectively. We provided a crosswise comparison to demonstrate the superiority of the proposed adder. Moreover, we implemented two previously studied configurable adders to evaluate power consumption, critical path delay, design area, and accuracy. We also evaluated the quality of these accuracy configurable adders in a real image processing application.

II. OBJECTIVE

The objective of the paper is to avoid the high power consummation and reduce path delay with in the adder circuit design. Approximate adders are widely being advocated for developing of hardware accelerators to perform complex arithmetic operations

III. STRUCTURE OF AN ADDER

The structure of the proposed 16-bit adder is shown in Fig 1. as an example. Four groups (CMHA3-0, CMHA7-4, CMHA11-8, and CMHA15-12) are used to prepare the P and G signals. Each group comprises four CMHAs

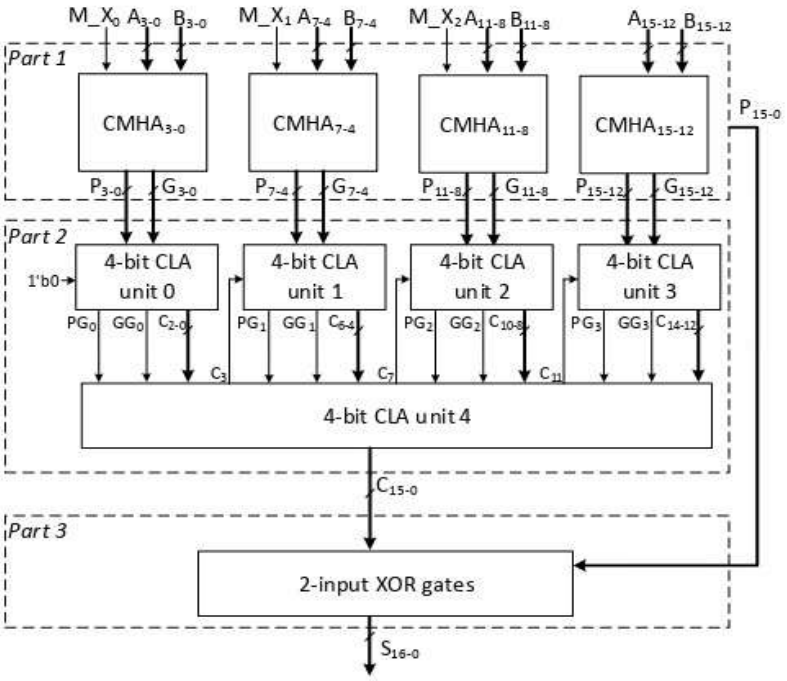


Fig. 1. Structure of the proposed 16-bit adder.

There is no mask signal for $CMHA_{15-12}$ in this example; therefore, accurate P_{15-12} (= A_{15-12} *XOR* B_{15-12}) and G_{15-12} (= A_{15-12} *AND* B_{15-12}) are always obtained. P_{15-0} and G_{15-0} are the outputs from Part 1 and are connected to Part 2. Note that the signal P_{15-0} is also connected to Part 3 for sum generation. In Part 2, four 4-bit carry look-ahead units (unit 0, 1, 2, 3) generate four PGs (PG_0, PG_1, PG_2, and PG_3), four GGs (GG_0, GG_1, GG_2, and GG_3), and 12 carries (C_{2-0}, C_{6-4}, C_{10-8}, and C_{14-12}) first, and then the carry look-ahead unit 4 generates the remaining four carries (C_3, C_7, C_{11}, and C_{15}) by using the PGs and GGs. C_{15-0} is the output of Part 2 and is connected to Part 3. The fifteen 2-input XOR gates in Part 3 generate the sum.

IV. PROPOSED ACCURACY-CONFIGURABLE ADDER

Typically, a CLA consists of three parts: (1) half adders for carry generation (G) and propagation (P) signals preparation,(2) carry look-ahead units for carry generation, and (3) XOR gates for sum generation. We focus on the half adders for G and P signals preparation in part 1. Let Consider an n-bit CLA; then outputs S_i and C_i obtained as follows:

$$P_i = A_i \oplus B_i,\ G_i = A_i \cdot B_i, \quad (1)$$

$$C_i = G_i + P_i \cdot C_{i-1} \quad (2)$$

$$S_i = P_i \oplus C_{i-1} \quad (3)$$

where *i* was denoted the bit position from the least significant bit. Note that owing to reuse of the circuit of A_i *XOR* B_i for S_i generation, here P_i is defined as A_i *XOR* B_i instead of A_i *OR* B_i. Because C_0 is equal to G_0 , if G_0 is 0, C_0 will be 0. From (2), we find that C_1 is equal to G_1 when C_0 is 0. In other words, if G_0 and G_1 are equal to 0, C_0 and C_1 will be 0. By expanding the above to *i* , C_i will be 0 when G_0, G_1, … , G_i are all 0. This means that the carry propagation from C_0 to C_i is masked. From (3), we can obtain that S_i is equal to P_i when C_{i-1} is 0.

4.1 CARRY GENARATION IN CLA

Output of full adder circuit

Sum $s_i = (a_i \oplus b_i) \oplus c_i$ -----1

carry $c_{i+1} = a_i.b_i + (a_i \oplus b_i).c_i$ ----2

Let's take

$P_i = A_i \oplus B_i$

$G_i = A_i \cdot B_i$

Now substitute P_i & G_i in equation 1 & 2

$$s_i = P_i \oplus c_i$$

$$c_{i+1} = G_i + P_i.c_i \quad \text{-----3}$$

Putting $i = 0, 1, 2, 3$ in Equation 3, we get

$$\begin{aligned} c_1 &= G_0 + P_0.c_0 \\ c_2 &= G_1 + P_1.G_0 + P_1.P_0.c_0 \\ c_3 &= G_2 + P_2.G_1 + P_2.P_1.G_0 + P_2.P_1.P_0.c_0 \\ c_4 &= G_3 + P_3.G_2 + P_3.P_2.G_1 + P_3.P_2.P_1.G_0 + P_3.P_2.P_1.P_0.c_0 \end{aligned}$$

From the above equations we understood the generation of the carry in carry look ahead adder

4.2 CARRY MASKABLE ADDER

A conventional half adder is shown in Fig. 2(a). A 2- input XOR gate is used to generate sum s and a 2-input AND gate is used to generate carry Cout. An equivalent circuit of a half adder is shown in Fig. 2(b). The dashed frame represents an equivalent circuit of a 2-input XOR gate. Since there is a 2-input NAND gate in the dashed frame, we reuse it and add an INV gate to generate the carry signal Cout. The outputs of the 2-input NAND and OR gates in the dashed frame are named u and w, respectively. Table 1 is the truth table for the equivalent circuit of a half adder.

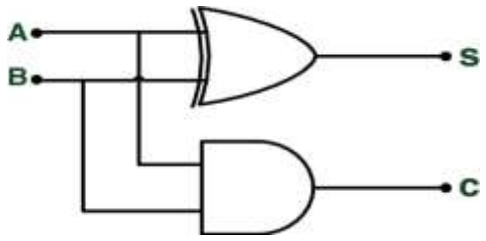


Fig. 2. (a) An accurate half adder

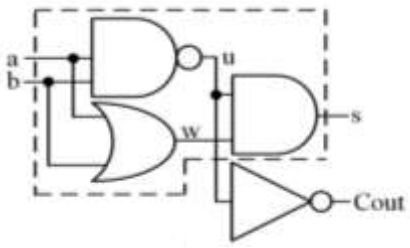


Fig. 2 (b) equivalent circuit of a half adder.

As shown in Fig. 2(b) and Table 1, when the internal signal u is 1, the sum s is equal to a *OR* b and the carry Cout is 0. This means that, if u is controllable and can be controlled to 1, the carry propagation will be masked and the sum s will be equal to a *OR* b. The sum s = a *OR* b is different from the accurate sum (=a *XOR* b) only when both a and b are 1. In other words, the sum s = a *OR* b can be considered as an approximate sum. The selectivity between the accurate and approximate sums can be achieved by a control signal, which is used to control u to be a *NAND* b, or to be 1.

Inputs		Internal signals		Outputs	
a	b	u	w	s	Cout
0	0	1	0	0	0
0	1	1	1	1	0
1	0	1	1	1	0
1	1	0	1	0	1

Table 1: Truth table for the equivalent circuit of a half adder.

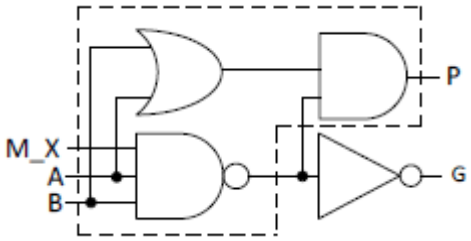


Fig. 3. A carry-maskable half adder.

From the perspective of approximate computing, if G is controllable and can be controlled to be 0, the carry propagation will be masked and S (=P) can be considered as an approximate sum. In other words, we can obtain the selectivity of S between the accurate and approximate sum if we can control G to be A *AND* B or 0. Evidently, we can achieve selectivity by adding a select signal. Figure 2(a) is a conventional half adder and Fig. 2(b) is a half adder to which the select signal has been added. Compared with named "M_X" as the select signal and use a 3-input AND gate to replace the 2-input one. When M_X = 1, the function of G is the same as that of a conventional half adder; when M_X = 0, G is equal to 0.

Consider the condition when the inputs Ai and Bi are both 1, when M_Xi = 1, the accurate sum Si and carry Ci will be 0 and 1 ({Ci , Si} = {1,0}); when M_X0 , M_X1 , … , M_Xi are all 0, Si is equal to Pi (= Ai XOR Bi = 0) as an approximate sum and Ci is equal to 0 ({Ci , Si} = {0, 0}) as discussed above. Here {,} denotes concatenation. This means that the difference between the accurate and approximate sum is 2. Toward better accuracy results for the approximate sum, we use an OR function instead of an XOR function for P generation when M_X = 0. Thus, the difference will be reduced to 1. A 2-input XOR gate can be implemented by using a 2-input NAND gate, a 2-input OR gate, and a 2-input AND gate. An equivalent circuit of the conventional half adder is shown in Fig. 3. This is called a carrymaskable half adder (CMHA). The dashed frame represents the equivalent circuit of a 2-input XOR (M_X = 1). We can obtain the following: P is equal to A XOR B, and G is equal to A AND B when M_X = 1; when M_X = 0, P is equal to A OR B and G is 0. Thus, M_X is named as a carry mask signal.

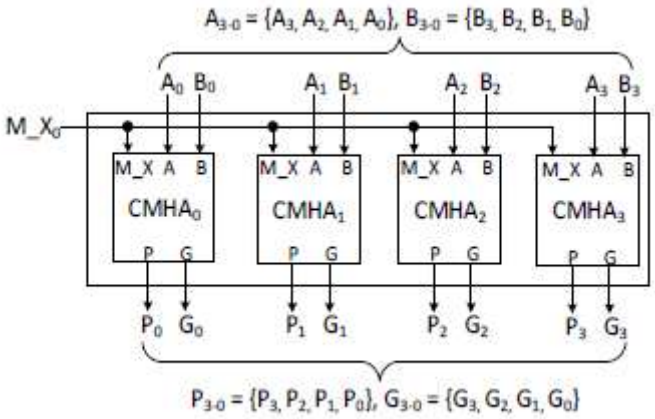


Fig. 4. Structure of a group with four CMHAs

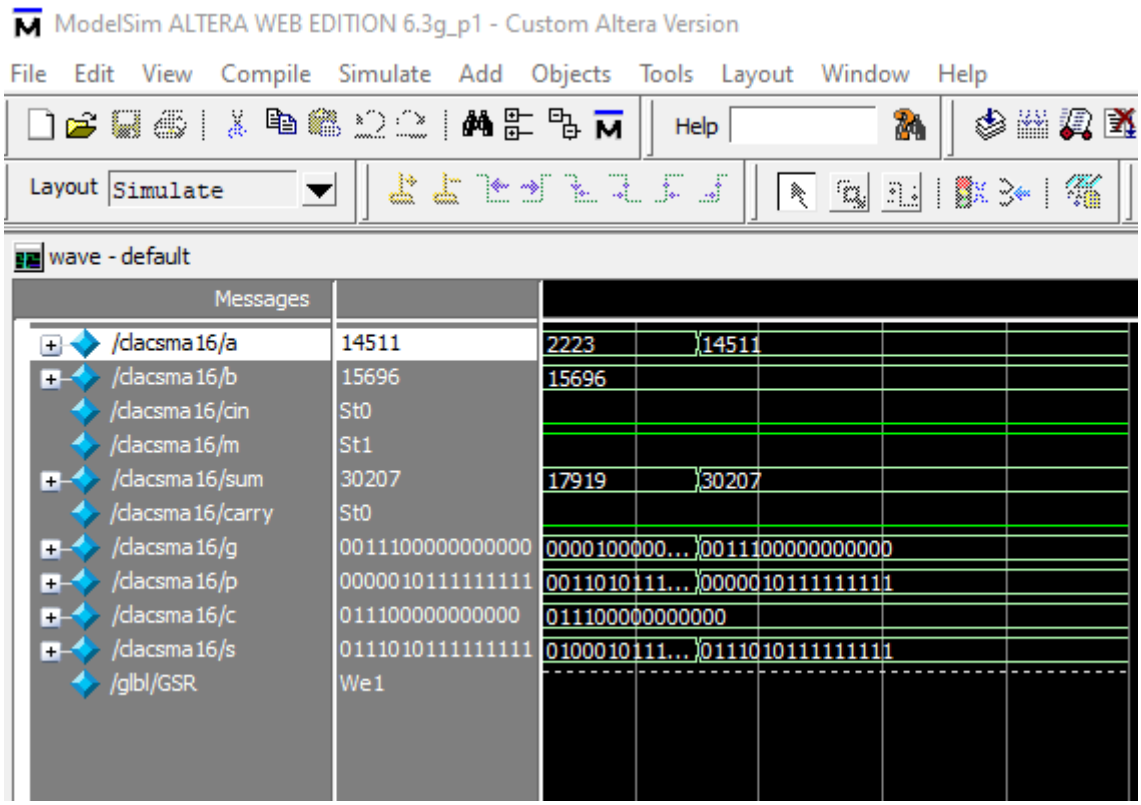
V. RESULTS AND DISCUSSION

5.1 ADDITION RESULT

If a=2223, b=15699, cin=0 and m=1 then sum=17922

If a=2223, b=15696, cin=0 and m=1 then sum=17919

If a=14511, b=15699, cin=0 and m=1 then sum=30207



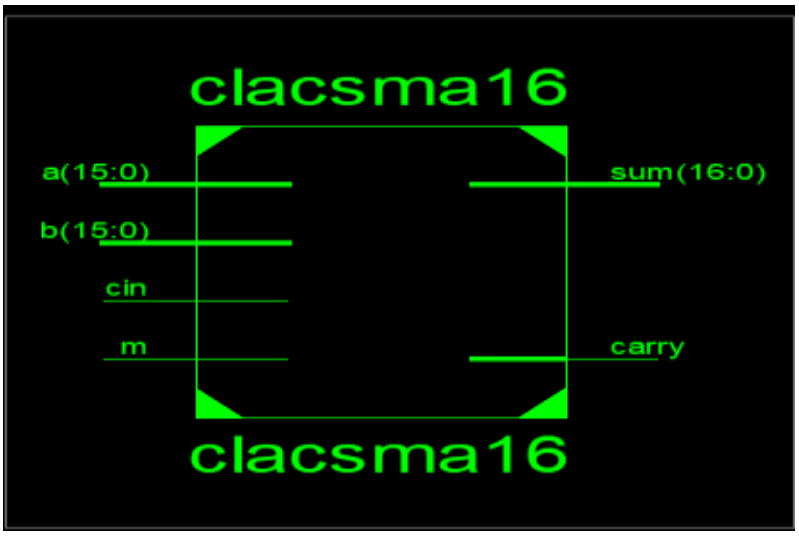
5.2. LOOK UP TABLES

clacsma16 Project Status			
Project File:	cla.xise	Parser Errors:	No Errors
Module Name:	clacsma16	Implementation State:	Synthesized
Target Device:	xc3s100e-5vq100	•Errors:	
Product Version:	ISE 14.7	•Warnings:	
Design Goal:	Balanced	•Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	•Timing Constraints:	
Environment:	System Settings	•Final Timing Score:	

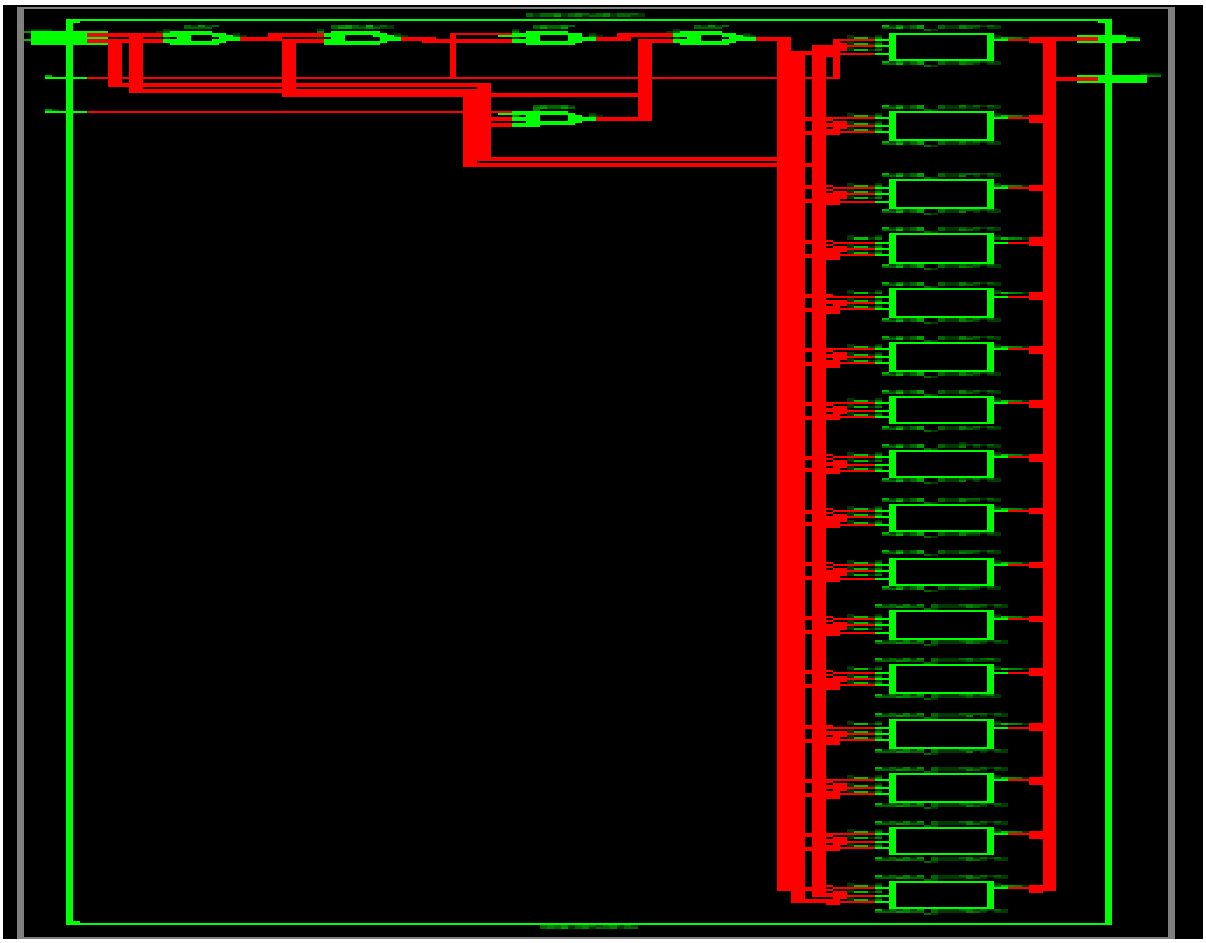
Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	1	960	0%
Number of 4 input LUTs	2	1920	0%
Number of bonded IOBs	4	66	6%

Detailed Reports					
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Tue 3. Mar 10:52:21 2020			
Translation Report					
Map Report					
Place and Route Report					
Power Report					
Post-PAR Static Timing Report					
Bitgen Report					

5.3 PIN DIAGRAM

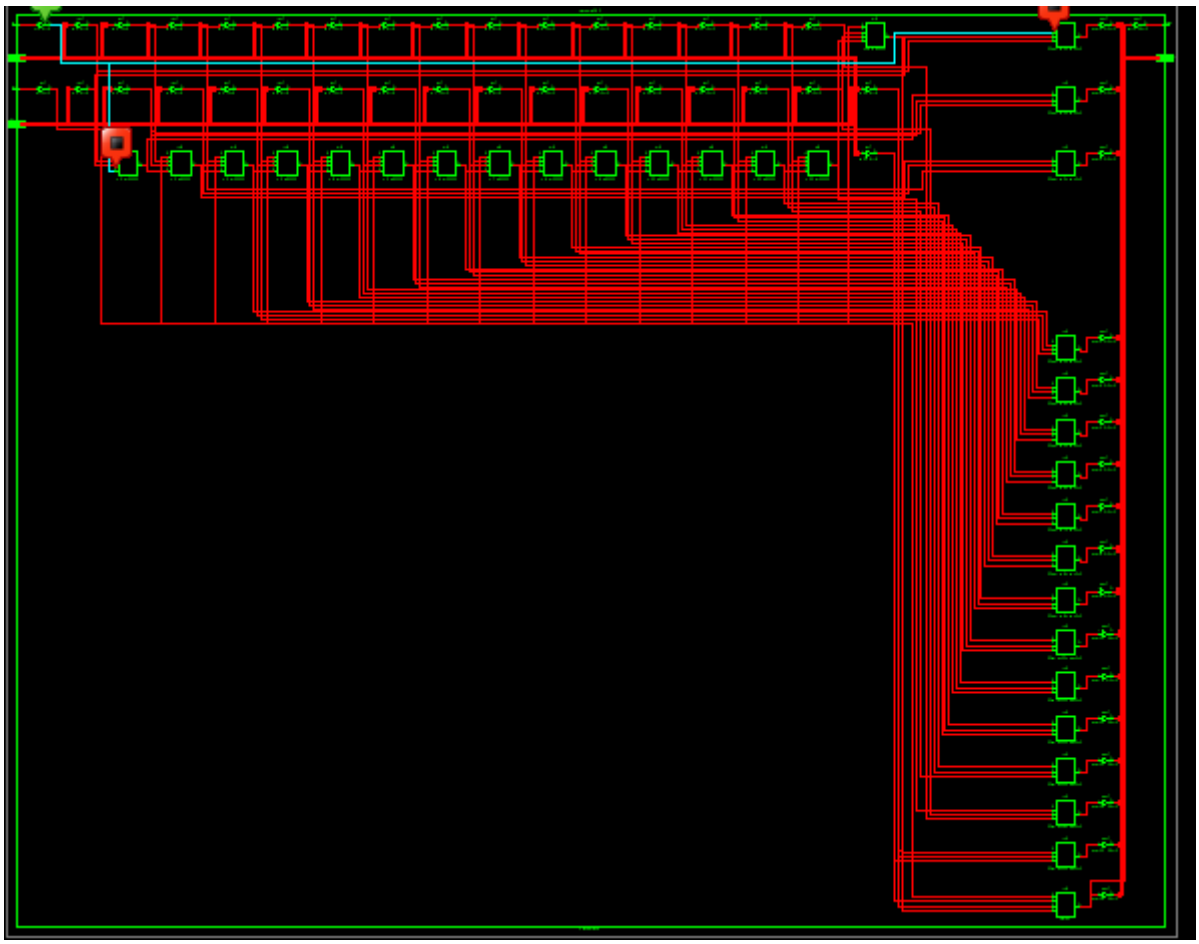


5.4 RTL SCHEMATIC



5.5 POWER ESTIMATOR

5.6 TECHNOLOGY SCHEMATIC



5.7 TIMING DETAILS REPORT

```
Timing Detail:
--------------
All values displayed in nanoseconds (ns)

=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 4 / 2
-------------------------------------------------------------------------
Delay:               5.776ns (Levels of Logic = 3)
  Source:            a (PAD)
  Destination:       carry (PAD)

  Data Path: a to carry
                                Gate     Net
    Cell:in->out      fanout   Delay    Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             2   1.106    0.532  a_IBUF (a_IBUF)
     LUT2:I0->O            1   0.612    0.357  carry1 (carry_OBUF)
     OBUF:I->O                 3.169           carry_OBUF (carry)
    ----------------------------------------
    Total                      5.776ns (4.887ns logic, 0.889ns route)
                                       (84.6% logic, 15.4% route)

=========================================================================


Total REAL time to Xst completion: 4.00 secs
Total CPU time to Xst completion: 3.86 secs
```

VI. CONCLUSION AND FUTURE

In this paper, an accuracy-configurable adder without suffering the cost of the increase in power or in delay for configurability was proposed. The proposed adder is based on the conventional CLA, and its configurability of accuracy is realized by masking the carry propagation at runtime. The experimental results demonstrate that the proposed adder delivers significant power savings and speedup with a small area overhead than those of the conventional CLA. Furthermore, compared with previously studied configurable adders, the experimental results demonstrate that the proposed adder achieves the original purpose of delivering an unbiased optimized result between power and delay without sacrificing accuracy. It was also found that the quality requirements of the evaluated application were not compromised

VII. REFERENCES

[1] S. Cotofana, C. Lageweg, and S. Vassiliadis, “Addition related arithmetic operations via controlled transport of charge”, IEEE Transactions on Computers, vol. 54, no. 3, pp. 243-256, Mar. 2005.

[2] V. Beiu, S. Aunet, J. Nyathi, R. R. Rydberg, and W. Ibrahim, “Serial Addition: Locally Connected Architectures”, IEEE Transactions on Circuits and Systems-I: Regular papers, vol. 54, no. 11, pp. 2564-2579, Nov. 2007.

[3] S. Venkataramani, V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, “Quality programmable vector processors for approximate computing”, 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), pp. 1-12, Dec. 2013.

[4] A. B. Kahng, and S. Kang, “Accuracy-configurable adder for approximate arithmetic designs”, IEEE/ACM Design Automation Conference (DAC), pp. 820-825, Jun. 2010

[5] R. Ye, T. Wang, F. Yuan, R. Kumar, and Q. Xu, “On ReconfigurationOriented Approximate Adder Design and Its Application”, IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 4854, Nov. 2013.

[6] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-Power digital signal processing using approximate adders", IEEE Transactions on Comptuer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 1, pp. 124-137, Jan. 2013.

[7] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-Inspired imprecise computational blocks for efficient VLSI implementation of Soft-Computing applications", IEEE Transactions on Circuits and Systems I: Regular papers, vol. 57, no. 4, pp. 850-862, Apr. 2010.

[8] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: modeling and analysis of circuits for approximate computing", IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 667-673, Nov. 2011.

[9] NanGate, Inc. NanGate FreePDK45 Open Cell Library, http://www.nangate.com/?page_id=2325, 2008

[10] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders", IEEE Transactions on Computers, vol. 62, no. 9, pp. 1760-1771, Sep. 2013.

[11] M. S. Lau, K. V. Ling, and Y. C. Chu, "Energy-Aware probabilistic multiplier: Design and Analysis", International Conference on Compliers, architecture, and synthesis for embedded systems, pp. 281-290, Oct. 2009.

[12] T. Yang, T. Ukezono, and T. Sato, "A Low-Power Configurable Adder for Approximate Applications", 19th International Symposium on Quality Electronic Design, Mar. 2018.