HARDWARE OPTIMISATION OF APPROXIMATE MULTIPLIER USING APPROXIMATE HIGH ORDER COMPRESSORS

[1]Mrs. V Jeyaramya M.E. (Ph.D), [2]Reethika J, [3]Priyatharisini E, [4]Pavithra R

[1]Associate Professor, Dept. of Electronics and Communication Engineering, Panimalar Institute of Technology, Chennai-600123,Tamilnadu,India.
[2]Student, Dept. of Electronics and Communication Engineering, Panimalar Institute of Technology,Chennai-600123,Tamilnadu,India.
[3]Student, Dept. of Electronics and Communication Engineering, Panimalar Institute of Technology,Chennai-600123,Tamilnadu,India.
[4]Student, Dept. of Electronics and Communication Engineering, Panimalar Institute of Technology,Chennai-600123,Tamilnadu,India.

Abstract - *The power optimization and reducing the components in the multiplier is a challenging part in a electronic device. Multiplier is a big deal in these devices as it is one the most complex and major source of power dissipation. Even though Approximate Computing provides internal inconsistency as it plays a major role in designing electronic applications. In order to prevail this issue, in this paper an explanation of the design of 16-bit Approximate multiplier with approximate high order compressor is designed. The simulation and synthesization of this approximate multiplier is done by using Xilinx ISE Design Suite 14.7.*

Keywords: Approximate, power dissipation, reliability, multiplier, Xilinx ISE.

1. INTRODUCTION

Digital signals play a major role in the design of most of the applications like audio compression, video compression, and processes wherein the Microprocessor and Digital Signal Processor(DSP) play a pivotal l role in handling the complications of digital signals. The operations like convolution, correlation and filtering are mostly computed by using Digital Signal. Of the electronic components Multipliers, Shifters and Adders play a notable role in executing these operations. Multipliers take more time and higher power than other components. The Approximate high order compressors are used to optimize the Approximate multipliers in order to increase their speed performance.

2. CATEGORIZATION OF MULTIPLIER

2.1 Approximate Compressor:

The significant role of the compressor is that it reduces the stages of the product and consumes low power along with low latency. High order compressors are used in-case of our design wherein these compressor produces efficient results as in-terms of power being utilised and in reduction of an LUT slices and providing speed of performance. The value of error rate(ER),error distance(ED) and normalized error distance(NED) are important factor to determine the final output in an approximate multiplier.

2.2 Approximate Multiplier:

These are the part of the circuit which exhibit high tolerance to inaccuracy which are advocated for energy-efficient computing in most of the applications. The prime reason for choosing approximate multipliers in our design is that (1) the type of full adder circuit used to construct the multiplier, (2) array or tree i.e. architecture of used to construct the multiplier, (3) placement of sub-modules in the main multiplier module.

3. LITERATURE SURVEY

Approximate compressors are widely studied and they are used in optimizing the multipliers for the reduction of errors in the design but this design gives unequal delay in the signal. **"Approximate Compressors for Error-Resilient Multiplier Design"** Zhixi Yang, Jie Han, Fabrizio Lombardi [1] These designs can also cause the increase in the count of the transistors being used and power dissipation. Recently **"A Low-Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery"**CongLiu, JieHan, Fabrizio Lombardi published wherein it consumes high area and so the quality of the image gets reduced. The design which was

published consumes high area and so the quality of the image gets reduced.

4. OBJECTIVE OF WORK

1. The proposed design is implemented in-order to reduce the partial product stages in multiplication.
2. The approximate multiplier along with the high order compressor helps to reduce the area, power, time delay and reduce the number of LUT's.

5. EXISTING SYSTEM

Three-Bit Stacking Circuit

The existing system is based on the method of "THREE BIT STACKING", this proposed system is perceived to be a 6:3 counter wherein all the "1" bits are grouped together by first stacking of all the input bits. Two 3-bit stackings are used in-order to output the 6-bit count. The symmetric stack techniques adds one or more extra layers of logic which is combined as 3-bit stacks.

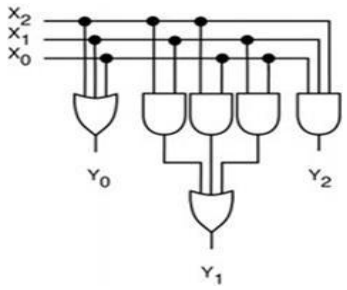


Fig-1: Three bit Stacker Circuit

The output formed from this circuit are Y0=X0+X1+X2

Y1=X0X1+X0X2+X1X2 Y2=X0X1X2

The output Y1 is the vital function which can be implemented using one CMOS gate.

TOP LEVEL MODULE

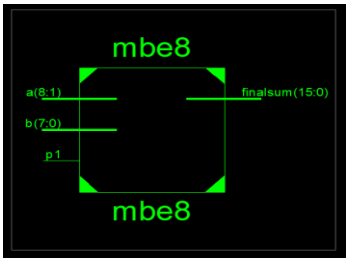


Fig-2: Top Level Module

TIMING ANALYSIS

```
Minimum period: No path found
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 20.671ns
```

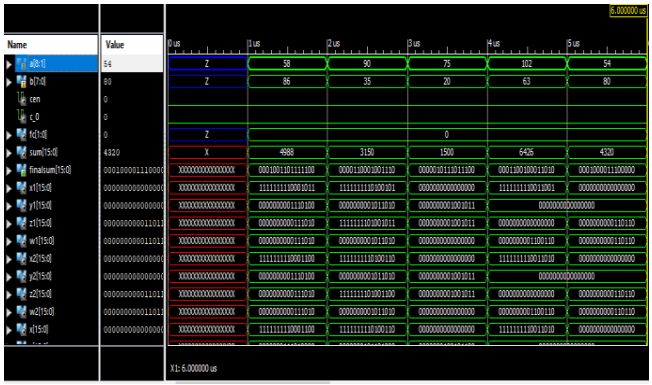
Fig-3: Timing Analysis

DEVICE UTILIZATION SUMMARY

mbe8 Project Status (02/21/2020 - 10:09:14)			
Project File:	SYMMETRIC_STAKING.xise	**Parser Errors:**	
Module Name:	mbe8	**Implementation State:**	Synthesized
Target Device:	xc6slx9-3tqg144	**• Errors:**	No Errors
Product Version:	ISE 14.7	**• Warnings:**	37 Warnings (37 new)
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 Slice LUTs	215	5720	3%
Number of fully used LUT-FF pairs	0	215	0%
Number of bonded IOBs	32	102	31%

Fig-4: Device Utilization Summary



OUTPUT WAVEFORM

Fig-5: Output Waveform

6. REQUIREMENTS OF APPROXIMATE MULTIPLIER

[I] Xilinx ISE is the major tool used for the implementation of the design which consists of the following fundamental steps: Design, Synthesis, Place and Route, Program.

[II] After these fundamental steps are accomplished the second part of simulation is the test bench simulation which is used to test the design by driving the inputs and observing the output to verify the design.

[III] Verilog is a Hardware descriptive language which is used in-order to describe the various propagation time and signal strengths. Since it takes less time to write large description of circuit in a short form, it is used in implementation design.

[1] BINARY MULTIPLIER

Two methods namely Partial product addition along with shifting or by using parallel multipliers the multiplication operation of two binary numbers can be performed.

OUTPUT WAVEFORM

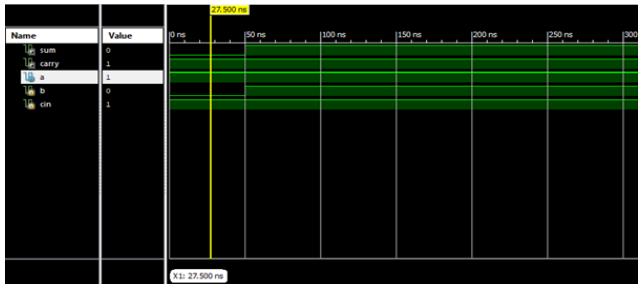


Fig-6: Output Waveform of Binary Multiplier

TIMING ANALYSIS

```
                                  Gate     Net
Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
----------------------------------------  ------------
 IBUF:I->O             16   1.222   1.109  a_i_IBUF (a_i_IBUF)
 LUT2:I0->O             4   0.203   1.048  t<2>1 (t<2>)
 LUT6:I0->O             3   0.203   0.995  f4/carry1 (c<4>)
 LUT5:I0->O             2   0.203   0.864  f7/carry1 (c<7>)
 LUT6:I2->O             2   0.203   0.981  f15/Mxor_sum_xo<0>1 (s<16>)
 LUT6:I0->O             2   0.203   0.721  f16/carry1 (c<17>)
 LUT6:I4->O             2   0.203   0.721  f22/carry1 (c<23>)
 LUT6:I4->O             1   0.203   0.827  f29/carry1 (c<30>)
 LUT6:I2->O             2   0.203   0.981  f35/Mxor_sum_xo<0>1 (s<36>)
 LUT6:I0->O             3   0.203   0.651  f36/carry1 (c<37>)
 LUT4:I3->O             2   0.205   0.981  f41/Mxor_sum_xo<0>1 (s<42>)
 LUT6:I0->O             2   0.203   0.981  f42/carry1 (c<43>)
 LUT6:I0->O             2   0.203   0.981  f46/Mxor_sum_xo<0>1 (s<47>)
 LUT6:I0->O             3   0.203   0.651  f47/carry1 (c<48>)
 LUT4:I3->O             2   0.205   0.981  f50/Mxor_sum_xo<0>1 (s<51>)
 LUT6:I0->O             2   0.203   0.981  f51/carry1 (c<52>)
 LUT6:I0->O             2   0.203   0.981  f53/Mxor_sum_xo<0>1 (s<54>)
 LUT6:I0->O             2   0.203   0.617  f54/carry1 (c<55>)
 LUT4:I3->O             1   0.205   0.579  f55/carry1 (out_15_OBUF)
 OBUF:I->O                  2.571          out_15_OBUF (out<15>)
----------------------------------------
Total                      24.087ns (7.453ns logic, 16.634ns route)
                                    (30.9% logic, 69.1% route)
```

Fig-7: Timing Analysis

DEVICE UTILIZATION SUMMARY

multiplier Project Status (11/19/2019 - 16:05:18)			
Project File:	Multiplier.xise	**Parser Errors:**	No Errors
Module Name:	multiplier	**Implementation State:**	Synthesized
Target Device:	xc6slx9-3tqg144	**•Errors:**	No Errors
Product Version:	ISE 14.7	**•Warnings:**	No 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 Slice LUTs	107	5720	1%
Number of fully used LUT-FF pairs	0	107	0%
Number of bonded IOBs	32	102	31%

Detailed Reports					
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Fri 24. Jan 15:26:13 2020	0	0	0
Translation Report	Out of Date	Wed 20. Nov 17:05:41 2019	0	0	0
Map Report	Out of Date	Wed 20. Nov 17:05:49 2019	0	0	6 Infos (0 new)

Fig-8: Device Utilization Summary

[2] BOOTH MULTIPLIER

The two binary number is multiplier using this method which is carried out by using an algorithm that multiplies two signed binary numbers using 2's complement.

OUTPUT WAVEFORM

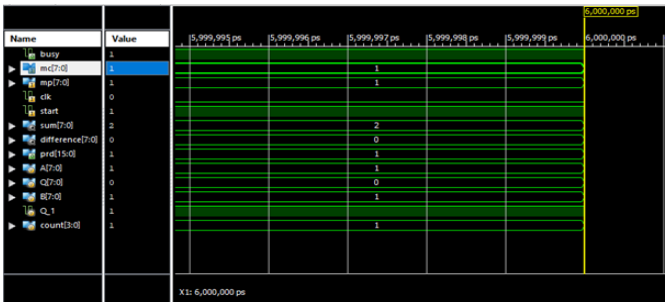


Fig-9: Output Waveform of Booth Multiplier

TIMING ANALYSIS

```
                                   Gate     Net
Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
----------------------------------------  ------------
  FDR:C->Q             3   0.447   0.650  count_3 (count_3)
  INV:I->O             1   0.206   0.579  busy1_INV_0 (busy_OBUF)
  OBUF:I->O                2.571          busy_OBUF (busy)
----------------------------------------
Total                      4.453ns (3.224ns logic, 1.229ns route)
                                   (72.4% logic, 27.6% route)
```

Fig-10: Timing Analysis

DEVICE UTILIZATION SUMMARY

wallace Project Status			
Project File:	Wallace.xise	**Parser Errors:**	No Errors
Module Name:	wallace	**Implementation State:**	Synthesized
Target Device:	xc6slx9-3tqg144	**• Errors:**	No Errors
Product Version:	ISE 14.7	**• Warnings:**	6 Warnings (6 new)
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 Slice LUTs	92	5720	1%
Number of fully used LUT-FF pairs	0	92	0%
Number of bonded IOBs	32	102	31%

Detailed Reports					
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Mon 9. Mar 17:58:46 2020	0	6 Warnings (6 new)	6 Infos (6 new)
Translation Report					
Map Report					

Fig-11: Device Utilization Summary

[3] WALLACE TREE MULTIPLIER

The multiplication of binary numbers using this multiplier is carried out by the reduction of the partial product matrix into two row matrix by half adder, full adder and carry save adder. A fast propagate adder are used to add these two rows.

OUTPUT WAVEFORM

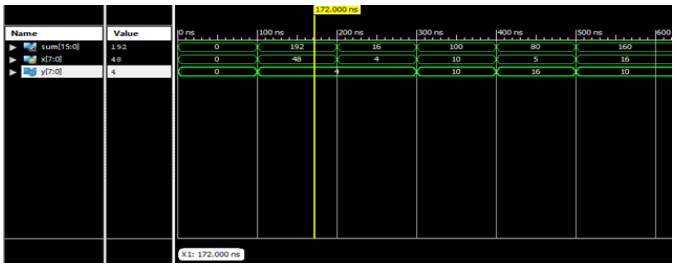


Fig-12: Output Waveform of Wallace Tree Multiplier

TIMING ANALYSIS

```
  LUT4:I1->O            1   0.205   0.808  a114/carry1 (c49)
  LUT6:I3->O            4   0.205   1.028  a100/Mxor_sum_xo<0>5 (su
  LUT6:I1->O            1   0.203   0.579  a78/carry1 (sum_15_OBUF)
  OBUF:I->O                 2.571          sum_15_OBUF (sum<15>)
----------------------------------------
Total                      19.706ns (6.440ns logic, 13.266ns route)
                                   (32.7% logic, 67.3% route)
```

Fig-13: Timing Analysis

DEVICE UTILIZATION SUMMARY

booth_multiplier Project Status (03/10/2020 - 11:04:46)			
Project File:	booth.xise	**Parser Errors:**	No Errors
Module Name:	booth_multiplier	**Implementation State:**	Synthesized
Target Device:	xc6slx9-3tqg144	**• Errors:**	No Errors
Product Version:	ISE 14.7	**• Warnings:**	17 Warnings (17 new)
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 Slice Registers	28	11440	0%
Number of Slice LUTs	38	5720	0%
Number of fully used LUT-FF pairs	20	46	43%
Number of bonded IOBs	35	102	34%
Number of BUFG/BUFGCTRLs	1	16	6%

Fig-14: Device Utilization Summary

7. APPROXIMATE MULTIPIER

7.1 Approximation of Carry

The equation can be implemented as modified half adder and modified full adder. The below Figure gives the logic of modified half adder and the logic of modified full adder, respectively. The approximate logic can be constructed for carry output of an high order approximate compressor using modified half adder and modified full adder. Approximate 5:2 compressor is used to obtain the carry output. Examples: When the number of input bits is 5 (i.e., n = 5), we can split the 5 input bits into 2 groups: one group includes X0, X1, X2, and also includes X3 and X4.

MODIFIED HALF ADDER

Fig-15: Modified Half Adder

MODIFIED FULL ADDER

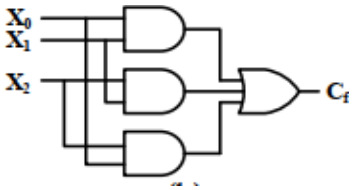


Fig-16: Modified full adder

(1).Carry output of approximate 5:2 compressor

Carry output of our approximate 5:2 compressor is:

Cf (X0, X1, X2) +Ch (X3, X4) + Ch (X0+X1+X2, X3+X4).

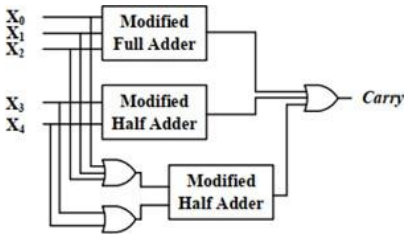


Fig-17: Modified Carry Output

7.2 Approximation of sum

We study the approximation of the logic of *Sum* output. Conventionally, the trees of XOR gates are used to produce the output Sum. However, compared with other logic gates, XOR gate often has larger design overheads. We use the logic gates in SAED 32nm cell library as an example. Table I tabulates the comparisons among OR gate, NOR gate, XNOR gate, and XOR gate. FromTable I, we find that XOR gate has the largest power, the largest area, and the largest delay. Thus, if we can replace XOR gates with other logic gates, all the design over heads (including the power, the area, and the delay) can be reduced.

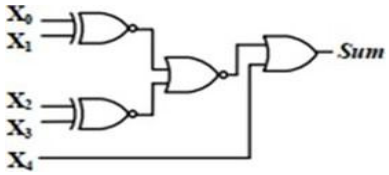


Fig-18: Modified Sum Output

7.3 RESULT

TOP LEVEL MODULE

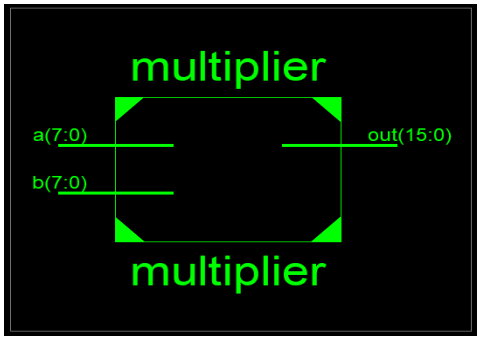


Fig-19: Top level Module

7.4 TIMING ANALYSIS

```
Minimum period: No path found
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 10.073ns
```

Fig-20: Timing Analysis

7.5 OUTPUT WAVEFORM

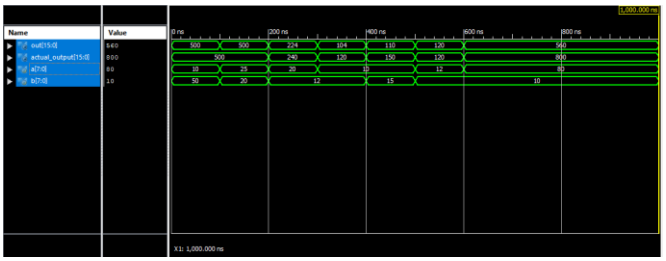


Fig-21: Output Waveform

DEVICE UTILIZATION SUMMARY

multiplier Project Status (02/21/2020 - 10:55:14)			
Project File:	highorder.xise	**Parser Errors:**	No Errors
Module Name:	multiplier	**Implementation State:**	Synthesized
Target Device:	xa7a100t-2Icsg324	**•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 Slice LUTs	69	63400	0%
Number of fully used LUT-FF pairs	0	69	0%
Number of bonded IOBs	32	210	15%

Fig-22:Device Utilization Summary

8. COMPARISON OF MULTIPLIERS

MULTIPLIERS	APPROXIMATE	BOOTH	SYMMETRIC STACKING	WALLACE TREE	BINARY
No of LUT's	62	75	215	92	107
No of IOB bonds	32	35	32	32	32
Power (mw)	1.5	2.0	3.1	2.2	2.8
Time delay (ns)	10.073	14.453	20.061	19.706	24.087

9. CONCLUSION

The approximate high order compressor architecture has been designed and synthesized using on Spartan6XC6SLX9 board and simulated in Xilinx ISE Design Suite 14.7. The performance of proposed Multiplier with high order compressor is compared with fast binary counter based symmetric staking multiplier. It can be inferred that high order compressor is faster and area efficient compared to binary counter based symmetric staking multiplier. In future the performance of the proposed multiplier can be improved and applied in applications like video and image processing.

10. REFERENCE

[1] Z.Yang, J.Han and F.Lambordi,"Approximate Compressor for Error-Resilient Multiplier Design", Proc. of IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems,2015.

[2] A.Momeni, J.Han, P.Montuschi and F.Lombardi , "Design and Analysis of Approximate Compressors for Multiplication" IEEE Trans. on Computers, vol 64, no.4, pp. 984-994,2015

[3] C. Liu, J. Han, and F. Lombardi, "A Low- Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery", Proc. of IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE),2014.

[4] G. Zervakis, et al., "Design-Efficient Approximate Multiplication Circuits Through Partial Product Perforation", IEEE Trans. on Very Large Scale Integration (VLSI) Systems, vol.24, no.10, pp. 3105-3117,2016.

[5] T. Yang, T. Ukezono, and T. Sato "A Low- Power High-Speed Accuracy Controllable Approximate Multiplier Design",Proc. Of IEEE Asia and SouthPacific Design Automation Conference (ASPDAC),2018.

[6] A. Cilardo, et al., "High-Speed Speculative Multipliers Based on Speculative Carry-Save Tree", IEEE Trans. on Circuits and Systems - I, vol. 61, no. 12, pp. 3426–3435,2014.

[7] J. Liang, et al., "New Metrics for The Reliability of Approximate and Probabilistic Adders", IEEE Trans. on Computers, vol. 62, no. 9, pp. 1760-1771, 2013.