MODIFIED PARTIAL PRODUCT GENERATOR FOR REDUNDANT BINARY MULTIPLIERS

PINAK CHOWDHURY P.G. Student, Digital Communication, MBE Society's College of Engineering, Ambajogai, Maharashtra, India

V.V.YERIGERI Assistant Professor, MBE Society's College of Engineering, Ambajogai, Maharashtra, India

ABSTRACT

A redundant binary is one of the popular methods in designing the high performance multipliers. The better modularity and the carry free addition make the system better in performance. The error correction takes place by means of error correction word. Authors have implemented and presented the redundant binary multipliers with partial product generator in this paper. The synthesis and simulation results are discussed in this paper. The speed enhancement is one of the key objectives of this operation.

KEYWORDS: Binary multipliers, redundant binary, partial product, error correcting word, etc.

INTRODUCTION:

Integrating the components was started in 19th Century by the scientists. Several electronic devices were developed and combined together to achieve the expected characteristics. In the initial days, the integration has started from connecting two transistors. The technology is developed now to very large and ultra large scale integration. The problem associated, when we go for more number of the devices is the speed of operation.

FPGA IMPLEMENTATION:

The logic blocks and two dimensional arrays are interconnected in FPGA system. The programming of the blocks helps in performing the functions as desired to complete the given task. If we need to implement a complex design (CPU for instance), then the design is divided into small sub functions and each sub function is implemented using one logic block.



Fig. 1. FPGA interconnections The FPGA system leads to the presence of the complete system on one chip.





This gives the better utilization of the logic block. The latches are used to hold the values of the function according to the input.

FPGA DESIGN FLOW





The design entry is the hardware interface where the designer can interact with the system. Converting the code to the acceptable format of the device takes place in the synthesis



This implementation stage mainly consists of a sequence of three steps

- ➤ Translate
- ≻ Map
- Place and Route

SYNTHESIS RESULT:



Fig.5. RTL schematic of Top-level of Proposed 32x32 multiplier using RBMPPG

55E Project Navigator (P.20131013) - D12016-20	017 Jeee Projects\2016 IEEE PRO	ECTS/J6V19/FM/RM size - (F8MPPG2 (RTL	u		00.
File Edit View Project Source Proces	ss Tools Window Layout	Help			- 0
□ 2 0 0 4 0 0 × 0 0	× * * * * * * * *	3 A 90 86 80 PR	• I 🕈 💡		
espi ⇔⊡ & ×	(N)				
👔 View: 👻 🌠 Inglementation 🔿 🚂 Simulation	152		1		
Hierarchy	24				
RM RM	-				
x2s500e-4fg320 x2s500e-4fg320 x2s500e-4fg320					
III IIIIIII HENPFOZ (HENPFOZS)	0				
5W	10				
3	DH				
2	4				
-	H				
L					
	Lie I				
No Processes Running	(A)				
Processes R8MPPG2 *	A.				
E Design Summary/Reports	0				
🗃 🗟 🎾 Design Utilities	0				
User Constraints	 (i) 				
View RTL Schematic	Do-				
View Technology Schematic					
Check Syntax	*				
Generate Post-Synthesis Sk., La					
Generate Programming File					
The Configure Transf Davies	The Design From The	4	met. (1.3		
sar - begn - nes - branes	a perditanities bit	nucescent [7] [70] Kei		HOMMAN (KILL)	
iew by Category					+08
Desi	ph Ubjects of Top Level Block			Properties of Instance: 00	
Instances Pins	*	Signals *	Name	" Value	
8 10 Madd yout not2000(63 8 🔥	REMPPG2	E 🔥 REMPPG2	Type Instance Name	booth_24verilog u0	
Console O Entra 1 Warrings 👸 1	Find in Files Results 🔠 Wey by	Category			
et view such that the entire contents is visible					164
					States and States
		9 9 2			· · · · · · · · · · · · · · · · · · ·

Fig.6. RTL schematic of internal block of Proposed 32x32 multiplier using RBMPPG



Fig.7. Technology schematic of Top-level of Proposed 32x32 multiplier using RBMPPG



Fig.8. Technology schematic of internal block of Proposed 32x32 multiplier using RBMPPG

ISE Project Navigator (P.20131013) - D.\2016-20	17 leee Projects\2016 IEEE	PROJECTS\16V19\RM\RM.xise = [RBMPPG2 (Tech1)]		
👷 File Edit View Project Source Proces		rout Help		- # ×
o el 🗙 🗇 🖓 😓 🖉 🚺	ы \varkappa 🎤 🔑 🎉 🧝	/ 3 🔁 🖱 🗆 🗢 / 19 🕨 × x 📌 💡		
Design ↔ □ # ×	101			
👔 View: 🛞 🎇 Implementation 💿 🔜 Simulation				
🚱 Hierarchy	🛌 🔛 LUT Dialog	AND A PERSONNEL CONTRACTOR		
E RM				
REAL REMPEGE (REMPEGEV)	<u> </u>	LUT3 96		
6	<u>~</u>	INET = 96		
	Schematic	Equation TruthTable Karnauch Map		
	D ^R			
x	a 1			
□				
	12 E			
No Processes Running	(A)			
Processes RBMPPG2	1.8			
BE Design Summary/Reports	0			
🔅 🎾 Design Utilities	0			
Oser Constraints	0		•	
View RTL Schematic	Po .			
Check Syntax				
Generate Post-Synthesis Si				
Implement Design Generate Programming File				o -
an Continue Ternet Desice				•
Start Cosign C Piles C Lbraries	2		BMPPG2 (Tech1)	
View by Category			and there are	+0 <i>8</i> ×
Desig	In Object		w3/zub[33].u0/Mxor_sum_xo	
Instances A Pins	00000		Value	
→ u23/zub[33].u0/cout1 → u23/zub[33].u0/Moor.s.	KDMPP		u23/zub(331.u0/Mixor.sum.xo	<0>1
In an and state of the state of	No. of the local diversion of the local diver	UK Help	~	•
🛄 console 🥌 crors 🔟 Warnings 🍇 P	TO INTO			1102402 1022721
				[108496,108656]
🕐 🖉 🖸 😒 💆	5 🦁 📄	9 🤏 🥒		 3/25/2017

Fig.9. Internal block of Proposed 32x32 multiplier using RBMPPG

6.4 SYNTHESIS REPORT

This device utilization includes the following.

- Logic Utilization
- Logic Distribution
- Total Gate count for the Design

The device utilization summery is shown above in which its gives the details of number of devices used from the available devices and also represented in %. Hence as the result of the synthesis process, the device utilization in the used device and package is shown below.

1 0								
RBMPPG2 Project Status (03/25/2017 - 15:30:54)								
Project File:	RM.xise		Parser Errors:			No Errors	No Errors	
Module Name:	RBMPPG2		Implen	Implementation State:			Synthesized	
Target Device:	xc3s500e-4fg32	10	•	• Errors:			No Errors	
Product Version:	ISE 14.7		• Warnings:			32 Warni	32 Warnings (0 new)	
Design Goal:	Balanced		•	Routing Results:				
Design Strategy:	Xilinx Default (u	nlocked)	•	Timing Constra	ints:			
Environment:	System Settings		•	Final Timing Sc	ore:			
			1					
Device Utilization Summary (estimated values)								
Logic Utilization	Used	Avai		ailable	able Utiliz		zation	
Number of Slices		2	2559	4656				54%
Number of 4 input LUTs		4	4561		9312			48%
Number of bonded IOBs			129 232			55%		
		Detailed Pe	norte					[-]
Depart Name	ports	Emans	Maunings		Infec			
Report Name	Status	us Generated		Errors	warnings		Intos	
Synthesis Report	Current	nt Sat Mar 25 15:30:52		0	32 Warnings (0	new)	0	
Translation Report								
Map Report								
RBMPPG2.v 🙁 🔀 RBMPPG2 (RTL1) 💽 🔀 RBMPPG2 (Tech1) 🗵								

Fig.10. Synthesis report of Proposed 32x32 multiplier using RBMPPG

JEE Project Navigator (P.20131013) - Dx2016-2017 leee Projects/2016 IEEE PROJECTS/16/19/RM/RM/size - [vtf_RBMPPG2.v]									
File Edit View Project Source Process	Tools Win	dow Layout Help			_ 8 X				
n a X 🖞 🖇 🖗 📲 📢 🗍	» 🏓	P 🔊 A R 🛛 🖉 🗟 H 🖬 P 🖉 🖉	🕨 🗹 🦿 👔						
Design ↔ 🛛 🗗 X	E 24								
🕫 Veur 🔿 🛱 Independation 🗿 🗐 Smulation	21	nodule vtf RBMPPG2;							
a Berried	26	-							
	27	// Inputs							
Hierarchy ,	n_ 28	reg [31:0] a;							
	29	reg [31:0] b;	mg [31:0] b;						
00 🕀 🖸 xc3s500e-4fg320	= 30	11 A							
W VU_RBMPPG2 (VU_RBMPPG2.v)	31	// Outputs							
	- 3Z	wire (63:0) yout;							
2	20	ALLE OVE,							
2	7 35	// Instantiate the Unit Under Test	(UUT)						
-	% 36	REMPPG2 uut (
	% 37	.a(a),							
-	38	.b(b),							
N D H D H D H D H D H D H D H D H D H D	39	.yout(yout),							
A no Processes Running	2 40	.ovf(ovf)			=				
Processes: vtf_RBMPPG2	- 41);							
8" 🕂 🞾 Nim Simulator	42								
Behavioral Check Syntax	43	initial pegin							
🕅 📲 Simulate Behavioral Model	45	a = 0.							
-	46	b = 0:							
w	47	#100;							
	48	a = 32'd24;							
	49	b = 32'd16;							
	50	#500;							
	51	a = 32'd56824;			-				
		11			÷				
🍃 Start 💐 Design 🚺 Files 🊺 Libraries	🖺 Design Su	mmary (Synthesized) 🗵 📋 RBMPPG2.v 🛛	🗟 RBMPPG2 (RTL1) 🛛	🙎 RBMPPG2 (Tech1) 🔝 📄 vtf_RBMPPG2.v 🛛					
view by Category					+- 0 8 ×				
Design	Objects of To	p Level Block		Properties of Instance: u23/zub[33].u0/Hxor_sum_xo1					
Instances A Pins		≜ Signals ≜	Name						
- ⊅ u23/zub(33).u0/cout1 🔄 🗄 🔒 RE	IMPPG2	🗄 📥 R8MPPG2	Туре	luß					
🔹 u23/zub[33].u0/Morg.s. 📮									
Canada 🗿 Errera 👔 Warrison 隆 Err	in Elas Dan Is	· · · · · · · · · · · · · · · · · · ·	UNITETATE	M					
👔 Console 🍟 Errors 🧘 Warnings 🦓 Price) in Hies Kesura	s 🔠 Wew by Category							
					Ln 50 Col 12 Verilog				
		音 👩 Q 🚬			3:38 PM				
	U 💙				3/25/2017				

Fig.11. Test Bench for Proposed 32x32 multiplier using RBMPPG

👔 ISim (P.20131013) - (Defaultwcfg*)	April 10 Cold State	a produce of	and -				- 5 - X
👳 File Edit View Simulation Window Layout	Help						_ <i>E</i> X
🗋 🌶 🖥 🕹 🐰 🖨 🖨 🗙 🖉	× 3880 ₽ ₽ ₽	11813	22 17	1 🖬 🖓 🖓	LOOUS 🛛 🌜 📋 🗔 Re	łaunch	
Instances and Processes 🛛 🖶 X Objects	+□5×	<i>p</i>					1,000.000 ns
Smulaton Obj	jects for vtf_PBMPPG2	<i>p</i>		n	200	100	
	444	8 Name	Value		200 ns	401 ns	50 m
b 🗟 vtf REMPPG2 vt Object Name	ne Value	P S youtesul	2592088584	<u> </u>	384		279205594
þ 🦉 gibi 👌 🔰 yout	(63:0) D000000000000		56824	() X	24		56824
lg of N ≥ 1914	0 000000000000000	(i) 1 (i) 1 (i)	45616		lő		45615
b 🛐 b[31:	0 000000000000000000000000000000000000	t.					
		2					
		+					
к <u>п</u>		r.					
🛓 Instances and Processes 📔 Source Files 🕡	п	1					
Menary	+-□ 8 ×	11					
2		11					
		8					
		_		X1: 1,000.000 ns			
		+		-	1		
			Default w	fg*			
Console					+ 🗆 🗗 X Complation	log + E] & X Find in Files Results ↔ □ & >
15in P.20131013 (signature 0x8eF4fb42)							RALIERO
This is a Full version of ISm. Time resolution is 1 os							No Search Results
Simulator is doing circuit initialization process.							
ISm>							
Console Breakpoints Search Results							
Uses unsigned decimal representation							Sim Time 1,000,000 p
🚯 🖉 🔉 😵	0 🗒 0	Q					- <mark>k</mark> ⊿ () 339 PM 3/25/2017

Fig.12. Simulated output for Proposed 32x32 multiplier using RBMPPG

CONCLUSION

The authors have carried out the synthesis and the simulation based analysis of the RBPP system. The designed system has helped for eliminating the ECW. The fast operating system is found suitable for the practical applications. The performance of the system is enhanced by implementing the carry free addition. The speed of the large systems is an important issue to be resolved. The synthesis and simulation results are discussed in this paper.

REFERENCES

- 1) Y. He and C. Chang, "A power-delay efficient hybrid carry-lookahead carry-select based redundant binary to two's complement converter," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 55, no. 1, pp. 336–346, Feb. 2008.
- 2) S. Kuang, J. Wang, and C. Guo, "Modified Booth multiplier with a regular partial product array," IEEE Trans. Circuits Syst. II, vol. 56, no. 5, pp. 404–408, May 2009.
- 3) J. Kang and J. Gaudiot, "A simple high-speed multiplier design," IEEE Trans. Comput., vol. 55, no. 10, pp. 1253–1258, Oct. 2006.
- 4) F. Lamberti, N. Andrikos, E. Antelo, and P. Montuschi, "Reducing the computation time in (short bitwidth) two's complement multipliers," IEEE Trans. Comput., vol. 60, no. 2, pp. 148–156, Feb. 2011.
- 5) Y. He and C. Chang, "A new redundant binary booth encoding for fast –bit multiplier design," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 56, no. 6, pp. 1192–1199, Jun. 2009.
- 6) Y. He, C. Chang, J. Gu, and H. Fahmy, "A novel covalent redundant binary booth encoder," in Proc. IEEE Int. Symp. Circuits Syst., 2005, vol. 1, pp. 69–72.
- 7) Y. Kim, B. Song, J. Grosspietsch, and S. Gillig, "Correction to 'a carry-free 54b54b multiplier using equivalent bit conversion algorithm'," IEEE J. Solid-State Circuits, vol. 38, no. 1, p. 159, Jan. 2003.
- 8) W. Rulling, "A remark on carry-free binary multiplication," IEEE J. SolidState Circuits, vol. 38, no. 1, pp. 159–160, Jan. 2003.
- 9) M. Ercegovac and T. Lang, "Comments on 'a carry-free 54b54b multiplier using equivalent bit conversion algorithm"," IEEE J. Solid-State Circuits, vol. 38, no. 1, pp. 160–161, Jan. 2003.
- 10) G. Dimitrakopoulos and D. Nikolos, "High-speed parallel- prefix VLSI Ling adders," IEEE Trans. Comput., vol. 54, no. 2, pp. 225–231, Feb. 2005.