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.

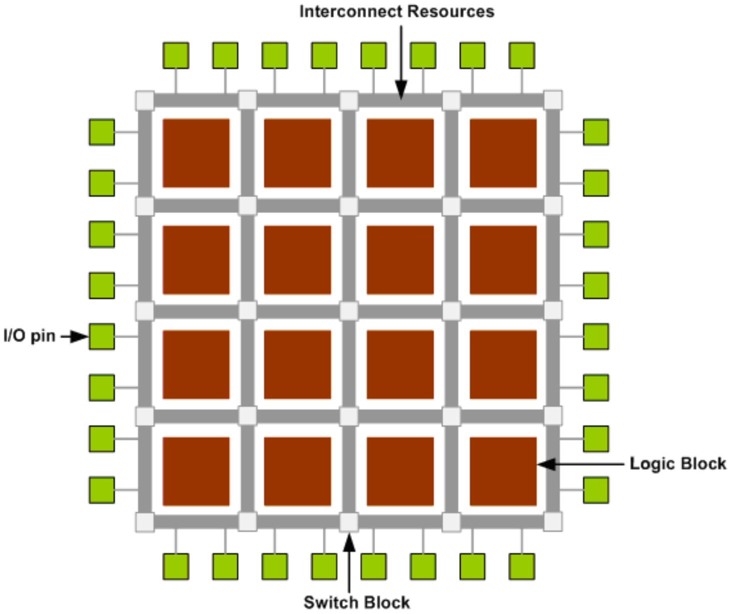


Fig. 1. FPGA interconnections

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

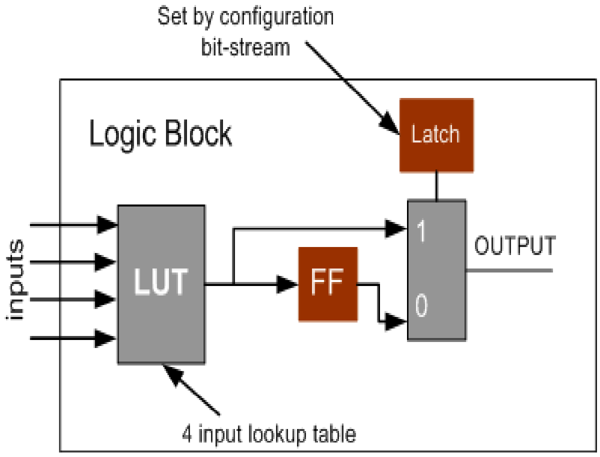


Fig.2. 4-input LUT based implementation of logic block

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

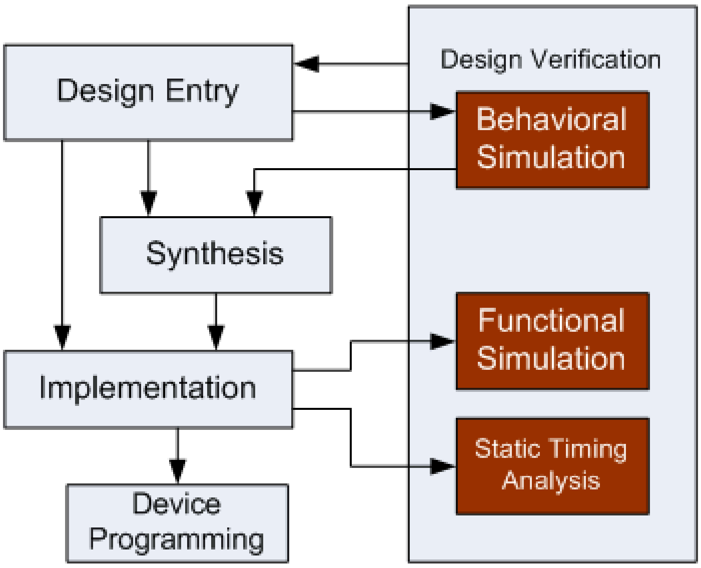


Fig. 3. FPGA Design Flow

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

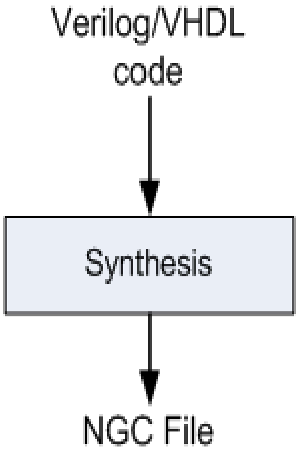


Fig.4. FPGA Synthesis

This implementation stage mainly consists of a sequence of three steps

- Translate
- Map
- Place and Route

SYNTHESIS RESULT:

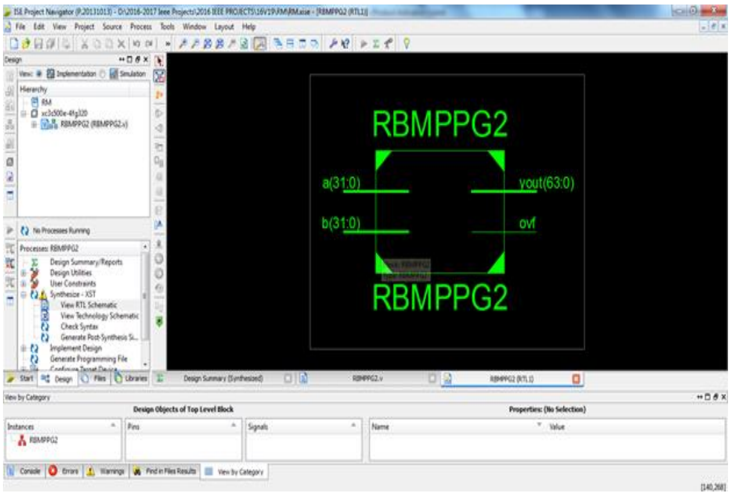


Fig.5. RTL schematic of Top-level of Proposed 32x32 multiplier using RBMPPG

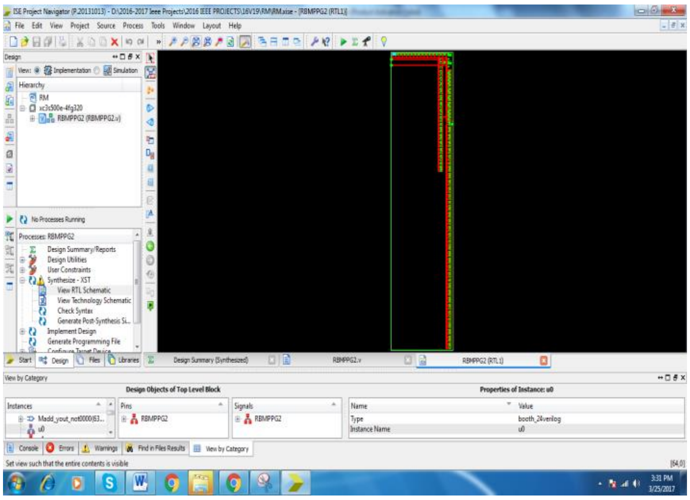


Fig.6. RTL schematic of internal block of Proposed 32x32 multiplier using RBMPPG

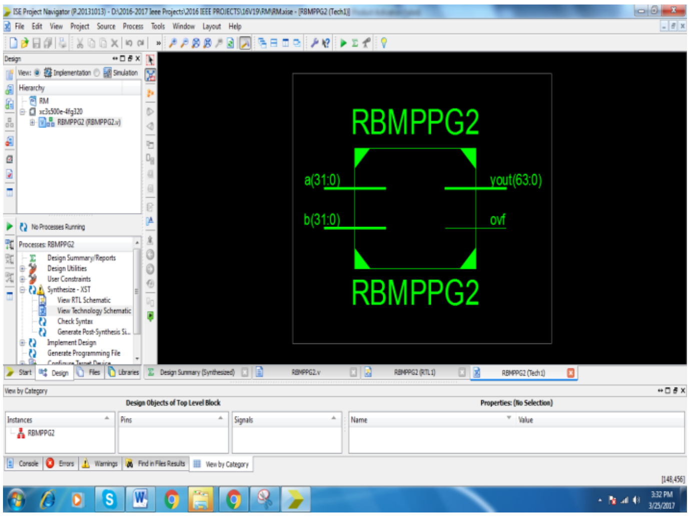


Fig.7. Technology schematic of Top-level of Proposed 32x32 multiplier using RBMPPG

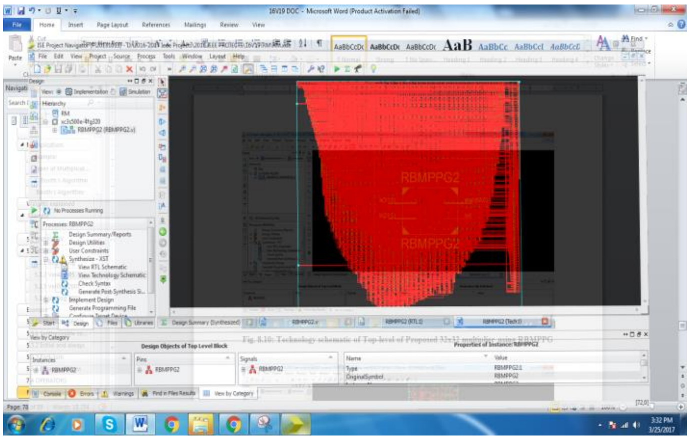


Fig.8. Technology schematic of internal block of Proposed 32x32 multiplier using RBMPPG

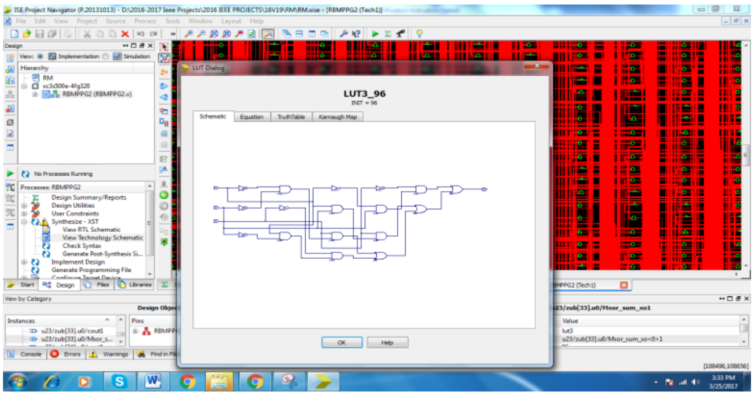


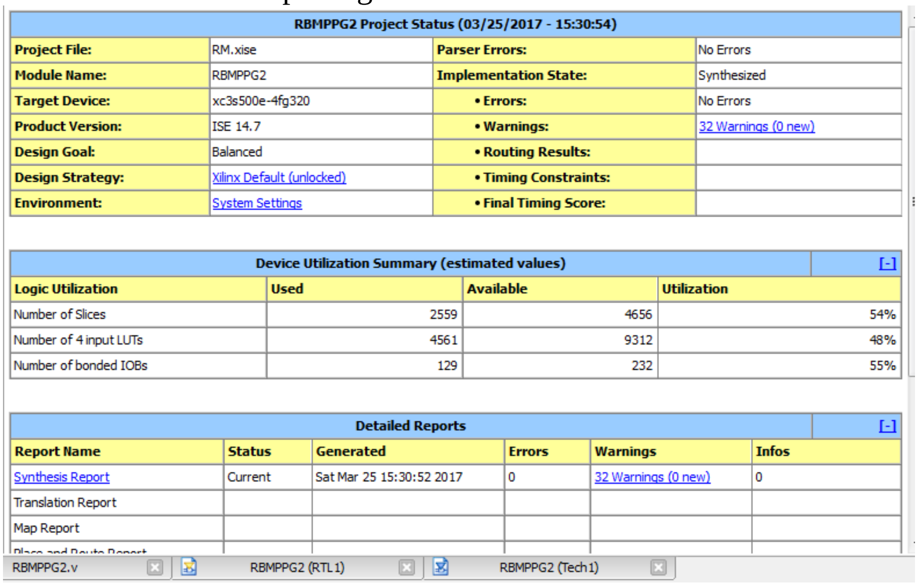
Fig.9. Internal block of Proposed 32x32 multiplier using RBMPPG

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.

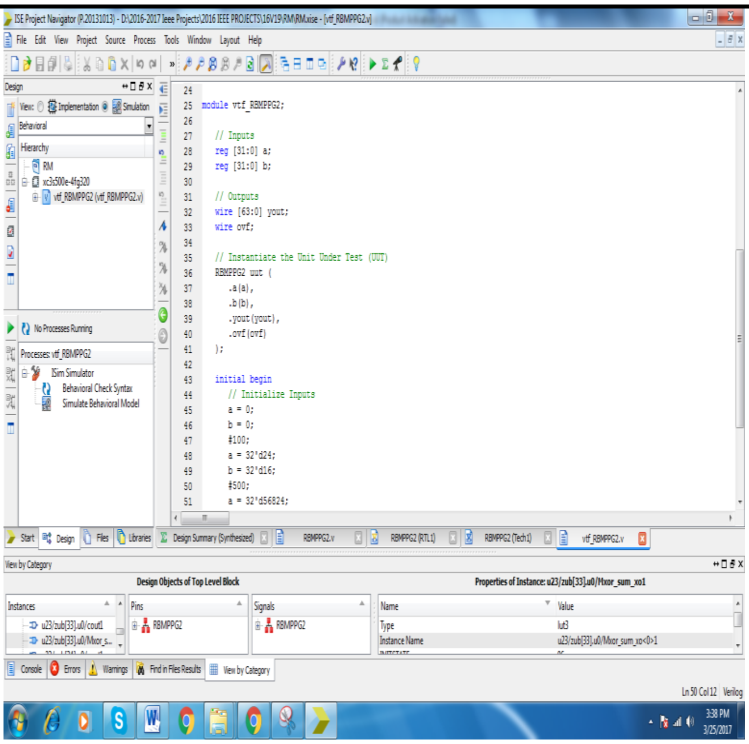


RBMPPG2 Project Status (03/25/2017 - 15:30:54)			
Project File:	RM.xise	**Parser Errors:**	No Errors
Module Name:	RBMPPG2	**Implementation State:**	Synthesized
Target Device:	xc3s500e-4fg320	• **Errors:**	No Errors
Product Version:	ISE 14.7	• **Warnings:**	32 Warnings (0 new)
Design Goal:	Balanced	• **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 Slices	2559	4656	54%
Number of 4 input LUTs	4561	9312	48%
Number of bonded IOBs	129	232	55%

Detailed Reports [-]					
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Sat Mar 25 15:30:52 2017	0	32 Warnings (0 new)	0
Translation Report					
Map Report					
Place and Route Report					

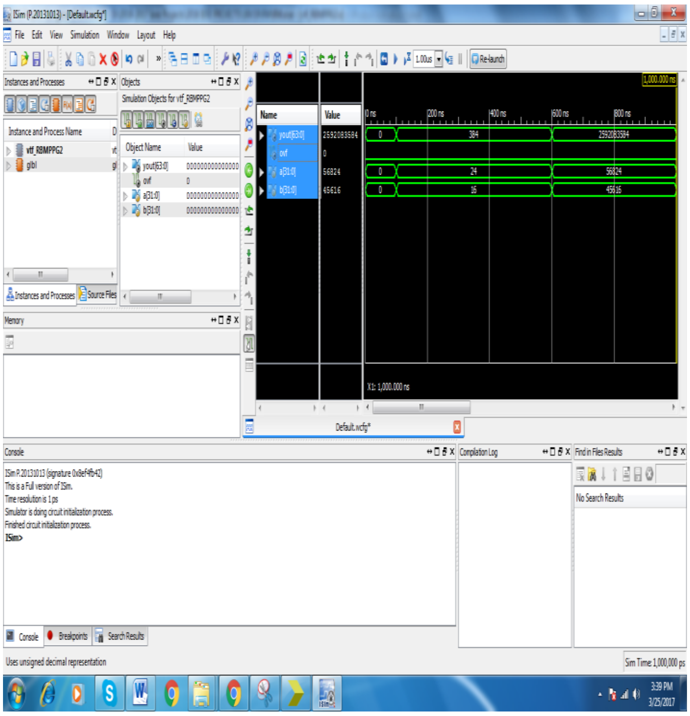
Fig.10. Synthesis report of Proposed 32x32 multiplier using RBMPPG


ISE Project Navigator (P.20131013) - D:\2016-2017 Ieee Projects\2016 IEEE PROJECTS\16V19\RM\RM.xise - [vtf_RBMPPG2.v]
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Behavioral
Hierarchy
RM
xc3s500e-4fg320
vtf_RBMPPG2 (vtf_RBMPPG2.v)
No Processes Running
Processes: vtf_RBMPPG2
ISim Simulator
Behavioral Check Syntax
Simulate Behavioral Model
Start
Design
Files
Libraries
Design Summary (Synthesized)
RBMPPG2.v
RBMPPG2 (RTL1)
RBMPPG2 (Tech1)
vtf_RBMPPG2.v
View by Category
Design Objects of Top Level Block
Properties of Instance: u23/zub[33].u0/Mxor_sum_xo1
Instances
Pins
Signals
Name
Value
Console
Errors
Warnings
Find in Files Results
View by Category
Ln 50 Col 12
Verilog
3:38 PM
3/25/2017

```
module vtf_RBMPPG2;

  // Inputs
  reg [31:0] a;
  reg [31:0] b;

  // Outputs
  wire [63:0] yout;
  wire ovf;

  // Instantiate the Unit Under Test (UUT)
  RBMPPG2 uut (
     .a(a),
     .b(b),
     .yout(yout),
     .ovf(ovf)
  );

  initial begin
     // Initialize Inputs
     a = 0;
     b = 0;
     #100;
     a = 32'd24;
     b = 32'd16;
     #500;
     a = 32'd56824;
```

Name	Value
Type	lut3
Instance Name	u23/zub[33].u0/Mxor_sum_xo<0>1

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



Object Name	Value
yout[63:0]	000000000000000
ovf	0
a[31:0]	000000000000000
b[31:0]	000000000000000

Name	Value
yout[63:0]	2592083584
ovf	0
a[31:0]	56824
b[31:0]	45616

```
ISim P.20131013 (signature 0x8ef4fb42)
This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initalization process.
Finished circuit initialization process.
ISim>
```

Uses unsigned decimal representation

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 bit-width) 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.