A NOVEL APPROACH FOR OPTIMIZING RTL POWER USING SYSTEM VERILOG ASSERTIONS

[1]Sangamesh Dundappa Ballur, [2]Nithya S
[1] PG Student [VLSI], [2]Assistant professor
[1] Dept. of ECE, SJB Institute of Technology
[2]Dept. of ECE, SJB Institute of Technology
Bangalore, Karnataka, India

Abstract— A novel of this project is based on the technique system Verilog assertions the power consumed by the RTL design can be optimized [4]. The designed technique supports the designer to boost of his RTL code achieving towards low power design. The designer written codes for the intended technique render to the proposed design specification and get the test bench of his own by coded integration technique [3]. The directive messages generated by the coded technique those are succeed to modify and enhance the code of the design to be optimize the power consumption of the design. The main idea behind to this approach is keep observing all the design signal. so, this approach is a dependent design technique, as it depends on the specifications of the design. This coded technique catches the targeted signals of intended input [1]. Then it shows the wrongly setting signals those may absorb extra wasted power. This approach can be applied and inserted in any of the verification framework as the universal verification methodology (UVM), to combine the optimized power features of this technique with the features of used framework. Through utilizing this technique the power consumed by the design is extremely optimized as it receives all remaining design signals that eat up extra wasted power. This project shows how to utilize the proposed technique about every design specifications and also afford technique code itself as a simple case study of RTL [3]. As well, it also assign resultant command or instructions messages and presents a difference between the power consumed before and after the alteration of the RTL code bestow to the command messages of the coded technique.

IndexTerms— System Verilog Assertions, Verification Environment, RTL, Test Cases, Universal Verification Methodology.

I. INTRODUCTION

With the extremely large growth in laptop, exclusive portable communication systems, and shrinking technology's evolution, the design technique in low power has been intensified by the research effort [1]. Now a days due to increasing in the count of portable applications needed architectures which utilize low power than ever before. Therefore low power eating architecture becomes the high priority candidates for microprocessor design and system components. Performing accurate and efficient power enhancement as early as possible in the flow of creating any design is important in creating a power optimized designs and essential for the successful low power designs [1]. Here the technique is concerned of performing accurate and efficient design power enhancement through being part of the design verification environment in the flow of creating RTL designs.

Objectives of the techniques are as follows:

- Suggesting a power optimization technique using system Verilog assertions.
- An accurate analysis of power optimization technique by a smooth case study to compare original code's power results with code that is optimized after implementing proposed technique.
- An overall view upon how these assertions were written to produce aimed test cases.
- Providing an approach of how to elite utilize power of any patchable code with respect to the power dissipated in signals activities.

II. PROPOSED RTL POWER OPTIMIZATION TECHNIQUE

Design verification is treated as the ultimate important operation in the SOC design flow as greater than 70% of the time is used on design verification. As the verification operation verify the design correctness, it may doesn't worry about other extra design code that doesn't perturb the design correctness [5]. This extra design code might cause eating more extra wasted power, since this code is not in the design details. But it doesn't perturb the design correctness. For example, if the design details tell that, only output signal(A) must be high when the input is set to high. But in real code implementation, the designer wrote such that, the pair of output signals (A) and (B) are high when input set to high. In the code implementation, signal (A) setting is see as a specification code but signal (B) setting is see as a additional code, this code not perturb the design correctness, but it eats some extra wasted power [3]. This technique makes use of the system Verilog assertions based verification to continuously watch the all design and every design signal verify functionally with respect to the design details. So this approach is a design dependent technique. Hence, in this technique we will elaborate the generation criteria of the technique and how to use it on any of the design specifications [5].

METHODOLGY

This technique explain the two dissimilar types of examination those are warning assertions and fault assertions by following assured creation steps, those are shown in fig 1.

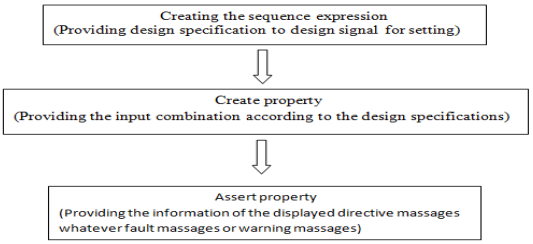

Fig 1. Flow of assertions creation.

Create Sequence Expressions

In any design, the functionality is depicted by mixture of multiple logical events. Those events are the clear Boolean expression that gets judge on the same clock edge or events that judge over a extent of time including multiple number of clock cycles [5]. System Verilog assertions give a key word to define these events say "sequence". In the sequence steps, the designer tags the signals design state at each single input combination bestow to the design detail where combination of each input sets some exact design to high state and the remaining to low state. Combination of every input signal has to be explained by of every input signal has to be explained by sequence of two different type which are operating sequence an state [5].

Operating Sequence:

After each setting state of the all design signals of each one input mixture in the state sequences. The engineer has to combine these state sequences into single operating sequence that will be tested during simulation. The binary operator (OR) is utilized to combine logically all other kind of state sequence of a confident one input combination to cover up all test summary explained in each state sequence. The operating sequence order attain goal outcome, when any state sequence order attain good outcome and will fail when whole state sequence fail [5].

Create Property

A number of sequences can be clubbed sequentially or logically to develop another complex sequence. SVA gives a keyword to show these complex sequential characters called property. In the step property, the input combinations are integrated by the designer with its operating sequence such that according to the specifications of the design [3]. In this integration, concurrent definitions will used by the designer which is calculated at clock cycles. Test expression is calculated at clock edges in concurrent assertions, based on examined value of the signals hooked. In SVA, delays of clock cycles are shown by a "##"sign and operations like implication represented by its different types [5]. Implication operators are off three types, non-overlap implication (|=>), overlapped implication (|->), and implication with fixed amount of delay on the consequent.

.Assert Property

It is a property during simulation it is going to be verified. This property asserted to catch the effect in the time simulation. SVA supports a key word named "assert" to test the property. In this step the property of every input combination is asserted by designer and it shows the displayed report directive massages [10]. A designer can print a success massege or custom error by utilizing the "action-block" in the statement assert. Certain system tasks are used to control during asperity of failing asserting in the action block. As ($fatal) can be used with fault massages and ($display) can be used with warning massages [3]. Fault assertions are must able to taking wrong code which disturb the design specifications and these assertions affect the design accuracy. So, fault assertions are used as directed test cases to verify signals. These fault assertion are shown in fig 1. excluding in property step. Operator (NOT) is used to develop forbidden services from happening, in property step. It means it is wanted that the property to be always false. The assertion fails if property is true. Thus fault massage directives are also displayed by fault assertions [5].

Warning Assertions:

Warning assertions take care for catching the extra code which does not alter the design accuracy, but consumes extra wasted power. Hence, these warning assertions developed by following warning assertion flow shown in fig.1. Warning assertions monitors direct warning massages that have directive information to be followed by the designer to modify and enhance the design code and reduce the design consumed power [5].

Fault Assertions:

Fault assertions are must able to taking wrong code which disturb the design specifications and these assertions affect the design accuracy. So, fault assertions are used as directed test cases to verify signals. These fault assertion are shown in fig 1 excluding in property step. Operator (NOT) is used to develop forbidden services from happening, in property step [5]. It means it is wanted that the property to be always false. The assertion fails if property is true. Thus fault massage directives are also displayed by fault assertions.

III. POWER ANALYZER FLOW CHART

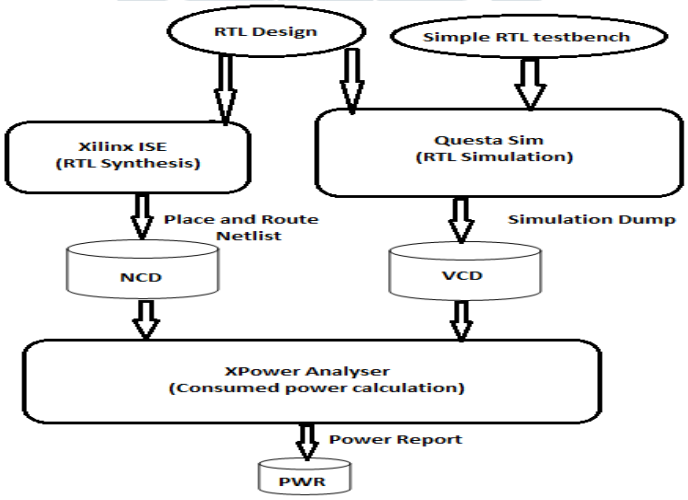

Fig 3. Power analyser flow chart

Follow the above flow chart you will get the power report. NCD (native circuit description file), VCD (value change dump file) both will give to Xpower analyzer you will get accurate power report. Software used namely Xilinx ISE 14.1 and Questa Sim 10.0C.

Case Study (DE_MUX):

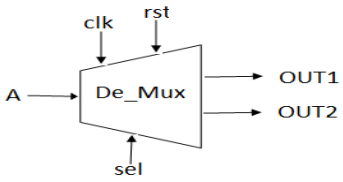

Fig 4. DE-MUX Schematic

Here DE_MUX consists of four inputs and two outputs. Inputs are A, clk, rst and sel. Outputs are out1, out2. If A=0, then out1 must be high or out2 must be high based on sel input. Suppose RTL designer writes unknowingly if A=0, then out1 and out2 both must be high based on sel line. Here unwanted switching activities exits. So here I m using SVA to reduce the unwanted switching activities those consume additional power. Hence by SVA I achieved power optimization to greater extent.

IV. RESULT

Applied Technique	Power consumed by the De_Mux code
Before applying the SVA	17mW
After applying SVA	14mW
Proposed SVA Approach(with ZYNQ)	10µW

Table 1. Comparison of Power consumption by the code.

In existing approach there also SVA are used to optimize the RTL code power consumption. In proposed technique I m using SVA with ZYNQ family (like Spartan). As shown in table 1, Before applying the system Verilog assertions code consumes 17mW power. After applying SVA code consumes 14mW. Proposed SVA approach with ZYNQ family it consumes 10µW.

V. CONCLUSION

This paper introduced a typical technique which handles the system Verilog assertions to optimize the power consumed by the given design. Here system Verilog assertions are utilized to continuously watch the all design signals since we have monitored each design signal can be set at any time. Hence this technique receives mistakes of the code in designed signals setting which consume added wasted power and directive messages are displays to correct the design code. Lax of the abstraction level of the design code, the proposed technique is a common methodology which can be enforced on any of the design code to increase it. The introduced technique can be involved in any verification framework to combine the power optimization characteristics of this technique with the characteristics of framework. Considering our unborn work we are going to build a computer aid design tool to faithfully generate the suggested types of assertions bestow to the design separations of the user to be hitched in the user verification atmosphere. Thus, the design does not wish to think about all likely cases and the CAD tool will automatically produce them.

REFERENCES

[1] David Howland and Russell. “RTL Dynamic Power Optimization for FPGAs” Tessier Department of Electrical and Computer Engineering University of Massachusetts Amherst, MA 01003. 978-1-4244-2167-1/08/$25.00 ©2008 IEEE.

[2] Nainala Vyagrheswarudu and Subrangshu Das and Abhishek Ranjan. “PowerAdviser: An RTL Power Platform for Interactive Sequential Optimizations” Texas Instruments Inc. Bangalore, India. Calypto Design Systems Inc. Noida, India. 978-3-9810801-8-6/DATE12/©2012 EDAA.

[3] Khaled Khalifa, Khaled Salah. “An RTL Power Optimization Technique Based on SystemVerilog Assertions” Alexandria University Alexandria, Egypt. Mentor Graphics Cairo, Egypt. 978-1-5090-1496-5/16/$31.00 © 2016 IEEE.

[4] Yeap, Gary K. Practical low power digital VLSI design. Springer Science & Business Media, 2012.

[5] S. Vijayaraghavan and M. Ramanathan. A Practical Guide for SystemVerilog Assertions. Springer, 2005.

[6] Kanika sahni, Kiran Rawat, Sujata Pandey and Ziauddin Ahmad. “Power Optimization of Communication System Using Clock Gating Technique” Amity University Noida Uttar Pradesh, India. Fifth International Conference on Advanced Computing & Communication Technologies 2015.

[7] Qi Wang, Sumit Roy. “RTL Power Optimization with Gate-level Accuracy” 555 River Oaks Parkway, San Jose 95125 2903 Cadence Design Systems, Inc. Calypto Design Systems, Inc Bunker Hill'Lane, Suite *208*, SantaClara 95054. *ICCAD'O3*, Novembcr II-l3,2003, San lose, California, USA.Copyright 2003 ACM 1-581 *13-762-11031001* I ... $5.00.

[8] IEEE Computer Society .“IEEE Standard for Verilog Hardware Description Language”, the Institute of Electrical and Electronics Engineers, Inc, Published 7 April 2006. Printed in the United States of America.

[9] Ashok B. Mehta. “SystemVerilog Assertions and Functional Coverage” Springer Science+Business Media , New York, 2014.

[10] Chris Spear. “Systemverilog For Verification” Springer Science+Business Media, LLC, 2006.

[11] http://www.ieee.org.in

[12] https://www.eda.ncsu.eda/wiki/tutorial:questa_system_verilog tutorial

[13] https://www.scribd.com

[14] http://www.spinger.com/978-1-4614-7323-7