International Journal of Scientific Engineering and Technology (ISSN : 2277-1581)
Volume No.1, Issue No.5, pg : 207-208 01 Nov. 2012

Employing Functional Analysis to Study Fault Models in VHDL

Venkatesh Kumar N[1], Mujeeb Ulla Jeelani[2], Amritha Mulay V[3], Anoop S Shandilya[4]

[1] Asst. Professor, SET, JAIN University, Bangalore, Freelance Consultant and Trainer.
[2]Software Engineer, 7Star Technologies, Bangalore.
[3]Software Engineer, Allied Tools and Electronics, Bangalore.
[4]Student, Signal Processing and VLSI, M.Tech, SET, Jain University, Bangalore.

Abstract:- In this paper, VHDL & functional analysis is used to model and to study the effect of faults on gate level circuits respectively. The model is developed via abstraction of industry standard single stuck line (SSL) faults into the behavioral domain and the effects of these faults on gate level circuits are discussed.

Keywords: Fault Models, Behavioral Modeling, VHDL, Functional Analysis, RTL Synthesis, Stuck Faults.

1. Introduction

The four prominent researchers Jacob Abraham, James Armstrong, Sumit Gosh and John Hayes have worked extensively in the field of functional and behavioral modeling, but their work were focused on functional fault models [1]. Armstrong have developed functional and behavioral fault models[2][3][4], Gosh proposed fault models based on the failure modes of language constructs[5] and finally Hayes proposed fault models leading to a fantastic class called induced faults[6]. The survey of high level fault modeling indicates that no widely accepted solution to the problem. Previous behavioral fault models don't have that much tenacity to link to the hardware which they attempt to describe. The demerits of the previous models are the main motivation for designing a new behavioral fault model based on a functional analysis of gate level circuits.

System behavior can be modeled using HDL. The synthesis tool understands the behavioral VHDL code as a description of an electronic circuit. Not all language constructs map directly to hardware and hence it necessitates to define a language subset. The VHDL behavioral models used in this research is based on IEEE draft standard for VHDL Register Transfer level Synthesis [7]. Previous work models used constructs *if-then-else* and *case*, such as *stuck-then/stuck-else* and *dead* clause, based solely on perturbing the language without a well defined link to the hardware [2][4][6]. The present standard of IEEE represents a subset of VHDL which is meant to ensure consistent synthesis of gate level net-lists. The key VHDL language constructs supported for behavioral modeling are:

1. *If* statements, *case* statements, *loop* statement
2. *constant, variable, signal*
3. Predefined VHDL operators.

Use of this subset is meant to enhance the portability of VHDL designs. Hence it is used here as the basis for defining higher level fault models which have a closer relation ship to resulting synthesized hardware.

2. Simple Design Example

In this work, *if* statement is used to select one of two input signals to be assigned to an output signal. The VHDL code was synthesized using Mentor Graphic's AutoLogic II. The resulting architecture is a multiplexer, which is shown in Fig.1

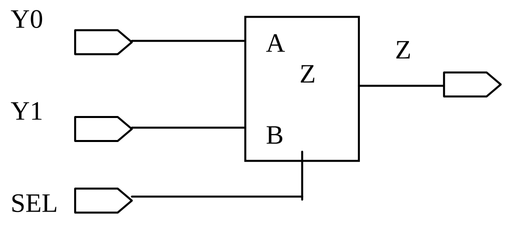

Fig.1 Multiplexer Architecture

3. Functional analysis

Multiple gate level implementations of the multiplexer were then evaluated. Sum of products (S) and product of sums (P) realizations including NAND and NOR only circuits were considered. A functional analysis was performed on the SSL gate level faults. After reductions for functional equivalence and fault dominance, the resulting fault tables form the basis for a functional fault model. Mandatory test vectors from the table provide complete coverage of the functional faults. A generalized functional fault model is therefore presented in Table 1. The functional faults have been designated SA,SB,PA and PB to indicate the origin of their mandatory test vector and the channel of the multiplexer which they corrupt. Testing based on the indicated vectors should provide complete coverage of gate level SSL faults over a broad range of implementations.

TABLE 1
Generalized Functional Fault Model

Test Vector	**SEL**	**Y1**	**Y0**	**Z**	**SA**	**PA**	**SB**	**PB**
	0	0	0	0				
Mandatory	0	0	1	1		0		
Mandatory	0	1	0	0	1			
	0	1	1	1				
	1	0	0	0				
Mandatory	1	0	0	0			1	
Mandatory	1	1	1	1				0
	1	1	1	1				

4. Development of a behavioral fault Model

Examination of the relationship between the generated functional fault model and the initial behavioral model will result in a behavioral fault model for the *if-then-else* construct. This final step in the abstraction of SSL gate level faults into the behavioral domain provides the link between lower and higher level fault models, which has been lacking in previous research.

Two of the functional faults affect the '*then'* clause. The S (sum of products) fault causes undesired activation of *Channel B*, while attempting to select *Channel A*. The functional fault SA, therefore, causes a corruption of *Channel A* by ORing it with *Channel B*. This fault caused by the *then* clause can be modeled by the *else* clause by performing ORing operation in the assignment statement.

Similarly, four set of faults are derived for the said example. These four faults form a fault model for the construct *if-then-else*.

5. Conclusions and Future scope

Novel approach has been employed to model, which are more closely linked to underlying hardware faults than previous fault models. The effects of the generalized sets of functional faults are abstracted into the behavioral domain by establishing a relationship between the higher level language construct and the lower level faults it should encompass. The fault modeling technique used is that of external corruption of the original constructs, rather than replacement or mutation of operators.

References

1. Abraram, J.A. and K.Fuchs, "Fault and Error Models for VLSI," *proceedings of the IEEE,* Vol. 74, No. 5 May 1986, pp 639-654.
2. Armstrong, F.S Lam and P.C Ward, "Test Generation and Fault simulation for Behavioral Models," *Performance and Fault Modeling with VHDL*, J.N Schoen, ed., Prentice Hall, Englewood Cliffs, NJ, 1992, pp. 240-303.
3. Armstrong J.R " B-algorithm: A Behavioral test Generation Algorithm," *Proceedings International Test Conf*, 1994, pp968-979.
4. J.R Armstrong, and Ward P.C "Behavioral fault Simulation in VHDL," *Proceedings 27th Design Conference,* June 1990, pp 587-593.
5. Gosh, S and Chakaraborty, " On Behavior Fault Modeling for Digital Designs," *Journal of Electronic Testing,* Vol.2 Kluwer Academic Publications, 1991, pp 135-151.
6. Hansen, M.C and J.P Hayes, "High-Level Test Generation using Physically Induced Faults," *Proceedings VLSI symposium,* May 1995, pp. 20-28.
7. IEEE *P1076.6/D1.12, Draft Standard for VHDL Register Transfer Level Synthesis,* VHDL synthesis Interoperability Working Group, IEEE, Piscataway, NJ, March 1998.