論文2003-40SD-12-6

VHDL 모델 검증의 효율적인 시간단축 방법

(Efficient Methods for Reducing Clock Cycles in VHDL Model Verification)

金 强 哲 *

(Kangchul Kim)

요 약

칩의크기가 증가함에 따라 VHDL 모델의 설계검증은 점점 어려워지고, 많은 시간을 소모하는 과정이 되고 있다. 최근에 VHDL 모델을 검증하기 위하여 베이지안 예측과 정지법(stopping rule)을 이용한 방법들이 소개되고 있다. 이 논문에서는 VHDL 모델을 검증하기 위하여 정지법을 사용할 때 클럭 사이클을 줄일 수 있는 2가지 방법을 제안한다. 첫 번째 방법은 세미랜덤변수를 정의하고, 정지법이 동작 중에 세미랜덤변수의 영역에 존재하는 데이터를 생략하여 정지점(stopping point)을 줄이고, 두 번째 방법은 정지법의 페이즈가 변화시에 베이지안 파라미터의 기존 값을 그대로 유지하여 클럭 사이클을 줄이는 방법이다. 제안된 방법의 효율성을 입증하기 위하여 12개의 VHDL 모델에 대하여 분기검출율에 관한 모의실험을 하였으며, 기존의 방법과 비교하여 분기검출율은 0.6% 줄었지만 25% 이상의 클럭 사이클을 줄일 수 있었다.

Abstract

Design verification of VHDL models is getting difficult and has become a critical and time-consuming process in hardware design. Recently the methods using Bayesian estimation and stopping rule have been introduced to verify behavioral models and to reduce clock cycles. This paper presents two strategies to reduce clock cycles when using stopping rule in a VHDL model verification. The first method is that a semi random variable is defined and the data that stay in the range of semi-random variable are skipped when stopping rule is running. The second one is to keep the old values of parameters when phases of stopping rule are changed. 12 VHDL models are examined to observe the effectiveness of strategies, and the simulation results show that more than about 25% of clock cycles is reduced by using the two proposed strategies with 0.6% losses of branch coverage rate.

Keyword : Verification, stopping rule, branch coverage, stopping point, semi-random variabl

I. Introduction

As VLSI fabrication technology has rapidly been increased, it is able to implement a single chip which comprises CPU, memory controller, bus controller etc. But the technologies of the design and the verification for complicated chips have not been developed rapidly, so design verification of behavioral models is getting difficult and has become a critical

* 正會員, 麗水大學校 컴퓨터工學科

(Computer Engeering, Yosu National University)

※ This work was done as a part of Information & Communication fundamental Technology Research Program supported by Ministry of Information & Communication in Republic of Korea.

接受日字:2003年3月11日, 수정완료일:2003年11月27日

and time-consuming process in hardware design. About 70% of the design effort in SoC(systems on chips), IP(intellectual property), ASIC, etc., has been consumed in verification process, and design and verification engineers work together from the beginning of IC design[1].

Hardware design is composed of various translation steps, that is, specifications, functional design, logic design, circuit and physical design, and reverse steps. Among the design steps, most of steps are automatically designed except of specification design. Since 1990, design technology using hardware description language has been increased rapidlyand most of complex chips have been designed by VHDL. VHDL is a complex concurrent language with typical models having many processes all conceptually executing at the time. So coverage metrics defined in software world have been borrowed and used in hardware design when the correctness of VHDL modeling is verified.

It is impossible to know that the design being verified is indeed functionally correct with 100% certainty. All of testbenches simulate successfully, but we can't make sure that there is no a function that has been forgotten to verify. That is why code coverage has to be used. Many different types of coverage criteria have been defined including statement, branch, block, expression, path coverage, etc[2-4]. And in many papers[2, 5], branch coverage is chosen as a good metric to verify hardware designs

A lot of clock cycles should be wasted to verify design by using coverage metrics because random test patterns have been used in complex VHDL modeling usually. Several examples showed that 5 billioninstruction simulation cycles were run to ensure fault-free chip before tapeout[6, 7]. Therefore the techniques that seek the optimal stopping point should be developed and it is necessary to take good stopping rule and strategy to reduce verification time and cost in a verification process.

This paper presents two strategies to reduce clock cycles when using stopping rule in VHDL model verification. In the first method, a semi-random variable is defined and the data that stay in the range of semi-random variable are skipped when stopping rule is running. The second one is to keep the old values of parameters when the phases are changed. More than about 25% of clock cycles is reduced by using the two proposed strategies with few losses of branch coverage.

In section 2, previous works are introduced, and proposed strategies are explained in section 3. Section 4 shows simulation results and discussions about the results. Finally, conclusion of this paper is described in section 5.

Ⅱ. Previous works

Many methods for testing software program have been developed[8-12]. Recently, the methods using Poisson distribution and Bayesian estimation were introduced by [8] and [12].

[8] suggested a compounded counting process using empirical Bayesian principles and introduced the stopping rule in verification of VHDL programs. The key idea was to combine two probability distributions, that is, the number of interruption and the size of interruption. The parameters of probability distributions were assumed random variables by using famous Bayesian estimation.

[12] observed the outcome of branch coverage at every clock cycle of testing and predicted the number of branch to be covered at time t. And the number to be covered was expressed by branch coverage random process X_t. X_t was divided to two probabilities, interruption N_t, where one or more new branches were covered at time t, and the size of the interruptions W_t. And the conditional distribution functions of the interruption occurrences, D_h is defined. [12] conducted a PMF(probability mass function) extraction experiments to estimate the best fitted distribution function at every discrete time t, then Poisson probability distribution function was chosen. Then W_t is defined to be a random process

distributed as a Poisson process with β_t

$$W_t \sim e^{-\beta_t} \frac{\beta_t^{w-1}}{(w-1)!} \quad (1)$$

where β_t is a random variable representing the parameter of Poisson distribution that should be estimated from the history of the simulation and is defined as $\beta_t = \beta g(t)$ for constant β and a decreasing function g(t) and $G(t) = \sum_{j=1}^{t} g(j)$. The probability of having an interruption during the testing process was estimated as p(t) for every discrete time t and p(t) was decomposed into a shape function f(t) and an amplitude value S_t, p(t)= S_t f(t), where the S_t values could be determined statically or dynamically based on the history of testing the behavioral model. And the Gamma function of β is given by

$$\Gamma(\beta;\gamma,r) = \frac{\gamma^r}{\Gamma(r)} e^{-\gamma\beta} \beta^{r-1} . \quad (2)$$

Then, statistical model for the branch coverage increase for a given history of verification was derived. And Bayesian analysis was done by calculating the likelihood function of the Bayesian parameter, β_t, as expressed in equation (3) and the coverage was expected at time t>T given the verification history like equation (4).

$$\hat{\beta} = \frac{r + x_t - n_t}{\gamma + G(t)} \quad (3)$$

$$E\{W_t \mid \vec{x}\} = 1 + \frac{r + x_t - n_t}{\gamma + G(t)} g(t) \quad (4)$$

Finally, the total number of branches to be covered at any future time t>T was predicted in equation (5).

$$E\{X_t \mid \vec{x}\} = x_T + \sum_{j=T+1}^{t} (1 + \frac{r + x_t - n_t}{\gamma + G(t)} g(j)) \phi f(j) \quad (5)$$

In most papers, a new pattern was generated at every simulation cycle and fed into a model[13, 14], but [12] represented a new verification strategy. When data bits had to be fixed for certain times, a pattern for a certain number of clock cycles was used, and 1, 2, 4, 6 clock cycles were remained for each phase stage. The mixed strategy of random testing improved the branch coverage[15].

Ⅲ. Proposed Strategies

The overall expected value of X_t is the sum of all the expected sizes of interrupts after time T as showed in equation (5). X_t and Wt have been assumed to be random variables for all clock cycles. If x_T is large in the equation (5), expected value of X_t is large, that is, the larger is getting sum of all coverage before time T, the longer is getting stopping point. Therefore sum of coverage before time T should be low to reduce the stopping point in time axis.

표 1. 클럭수에 대한 전체 분기검출률
Table 1. Total branch coverage vs. cc.

S#t	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
1	35	35	58	47	47	69	58	47	35	58	49	93
2	53	92	111	94	94	153	111	94	92	111	92	166
3	54	95	123	102	102	156	123	102	95	123	127	218
4	58	111	123	133	133	178	123	133	111	123	129	228
5	68	140	126	145	145	210	126	145	140	126	132	231
6	68	143	139	146	146	214	139	146	143	139	136	238
7	68	148	142	146	146	216	142	146	148	142	139	239
8	68	151	170	146	146	222	170	146	151	170	139	239
9	68	151	172	154	154	222	172	154	151	172	144	244
10	68	152	182	159	159	222	182	159	152	182	148	249

The values of random variables are not known with certainty, that is, we only know a set of possibletime and probability of the random variables. In this paper, semi-random variable is defined as a variable of which the value is certainly not known, but all values of the variable are greater than a referenced value. Table 1 shows total values of branch coverage for ten simulation times from the starting point for 12 sample VHDL models. Every

branch coverage for a few clock cycles from t=1 is large. Table 2 shows branch coverage for every clock cycle. For example, if the referenced value is 6, branch coverage is 35 at t=1 and 18 at t=2 respectively for S1, and each branch coverage is greater than 6 from t=1 to t=5 respectively for S5.

표 2. 매 클럭에 대한 분기검출율
Table 2. Branch coverage for every cc.

S# / CC	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
1	35	35	58	47	47	69	58	47	35	58	49	93
2	18	57	53	47	47	84	53	47	57	53	43	73
3	1	3	12	8	8	3	12	8	3	12	35	52
4	4	16	0	31	31	22	0	31	16	0	2	10
5	0	29	3	12	12	32	3	12	29	3	3	3
6	0	3	13	1	1	4	13	1	3	13	4	7
7	0	5	3	0	0	2	3	0	5	3	3	1
8	0	3	28	0	0	6	28	0	3	28	0	0
9	0	0	2	8	8	0	2	8	0	2	5	5
10	0	1	10	5	5	0	10	5	1	10	4	5

We can know that values of W_ts are comparatively large for a few clock cycles from the beginning time even though the exact values of W_ts can not be predicted. W_t can be divided into two time regions like equation (6), that is, W_{ts} is considered as semi-random variable in the first region and W_{tr} as random variable in the other region.

$$W_t = W_{ts} + W_{tr} \tag{6}$$

W_{ts} in the first part of t axis is so large that we can skip the data in the semi-random variable when stopping rule is running because W_ts are not predicted by Poisson's distribution function in the range. Semi-random variable is of consequence because the total number of branch coverage in the semi-random variable reaches nearly half of total coverage though the first region has a few clock cycles. So it is very important to make a reasonable decision of the referenced point in which the

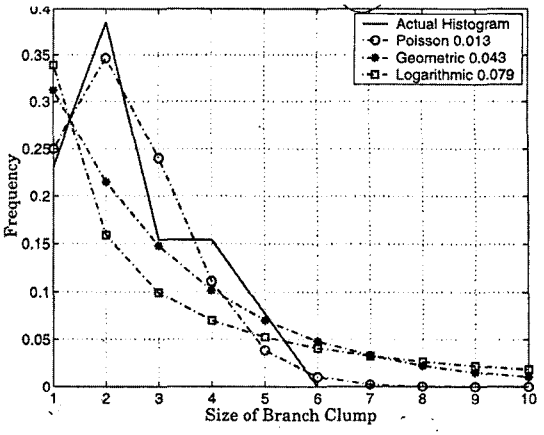

그림 1. S7에 대한 Wt의 히스토그램
Fig. 1. Histogram fitting example of Wt for S7.

referenced value occurs at first time at time axis because it has a great effect on stopping point.

The dashed line shows the probability of W_t for S7 in Fig. 1.[12] If the value of W_t is greater than 6, the probability is quite low. We can recognize that the probability that values of W_t are greater than 6 in Fig 1 is very small, but the probability in the semi-random variable is always 1. The clock cycle of which branch coverage is less than 6 at first time has been chosen as a referenced point which divides between semi-random and random variable. Depending on behavioral models and test patterns, the value of branch coverage is varied so that the time of the referenced point can't be fixed, which is dynamically decided while simulation is running.

If the branch coverage in region W_{ts} are skipped, stopping point will become short because x_T get lower in equation (5). As stopping point is getting short, we have to consider the problem that total branch coverage is getting low. But random test patterns are used in most behavioral model verifications before testing chip, and we know branch coverage rate abruptly drops as the number of random test pattern is increasing since a saturation point that the slope of branch coverage rate become below 45°. So we can predict that total branch coverage does not decrease abruptly.

The values of parameters gained at the previous

phases in equation (5) until time T have been thrown away and they are reset when a new phase get started in[12]. But Bayesian model predicts the expected branch coverage using the previous branch coverage. The overall expected value of X_t at any time t>T given the verification history up to time T is the sum of all the expected sizes of interruptions after time T, so the previous data which got in the previous phase have been used as initial values at the next phase in this paper.

Applying new verification strategies to the stopping rule of Bayesian model can reduce clock cycles and improve the performance of behavioral model verification.

Ⅳ. Simulation results and discussions

12 VHDL models are examined to compare the effectiveness of strategies with stopping rule of[12]. Table 3 shows the information of sample VHDL models.

표 3. 샘플 프로그램
Table 3. Sample programs.

Model	Description	# of code line	#of branch
S1	16 megabit byte-wide top boot 150ns	1880	373
S2	CMOS syncBIFIFO 256x36x2	4657	251
S3	SyncFIFO with bus-matching 1024x36	5015	302
S4	SyncFIFO 2048x36	4667	225
S5	SyncFIFO 2048x36	4710	225
S6	CMOS syncBIFIFO 1024x36x2	4949	296
S7	SyncFIFO with bus-matching 1024x36	4963	302
S8	SyncFIFO 2048x36	4777	225
S9	CMOS syncBIFIFO 512x36x2	4752	251
S10	SyncFIFO with bus-matching 512x36	4973	302
S11	SyncBIFIFO with bus-matching 512x36x2	5498	399
S12	SyncBIFIFO with bus-matching 1024x36x2	5770	470

Table 4 shows simulation results of 3 kinds run by Mentor QuickVHDL simulator, and stopping rule is run with branch coverage(BC) of each clock cycle(CC). In Table, SB is the result for static Bayesian estimator in[12]. In SBF, the branch coverage of which number is larger than 6 from the starting point are skipped, and the values of previous parameters in equation (5) are used whenever phases are changed under the same condition of SB. In SB30, the constraint that stops the simulation in a current phase whenever branch coverage for 30 continuous test patterns are zero is added to SBF. SBF reduces 24.6% of clock cycles and 0.6% branch coverage, and SB30 reduced 59.1% and 1.5% compared to SB respectively. Most branches are detected in phase 1 and coverage rate is decreased as number of clock cycles is increased. We can know that an increase of coverage rate gets slow after the rate is saturated. If constraint that makes simulation quit after a saturation point is added in a stopping rule, we know that the constraint can reduce a lot of clock cycles with least reduction of branch coverage. SB30 and SBF have 3 and 2 additional constraints compared to SB respectively, but they get better results than SB.

표 4. SB, SB30, SBF에 대한 모의실험 결과
Table 4. Simulation results of SB, SB30 and SBF.

	SB[12]		SB30		SBF	
	CC	BC	CC	BC	CC	BC
S1	1854	128	530	128	1460	128
S2	631	199	493	204	1461	206
S3	808	232	495	231	485	231
S4	673	192	485	192	482	192
S5	789	195	485	195	482	195
S6	2249	273	490	247	489	249
S7	809	232	495	231	496	231
S8	2181	208	649	203	1456	208
S9	631	199	493	204	1461	206
S10	808	232	495	231	496	231
S11	1982	245	596	245	1460	247
S12	2080	364	628	348	1462	360
SUM	15,495	2,699	6,334	2,659	11,690	2,684

It is difficult to get 100% of branch coverage because random test patterns are applied to the simulator. The effect of semi-random variable is hidden because some of the simulation results have 30 zero branch coverage so that the simulation is

stopped by 30 zero constraints. Simulation results of Table 4 that don't have zero branch coverage for 30 continuous test patterns for phase 4 are chosen in Table 5 again to consider the effect of the proposed strategies. Stopping point of simulation is decided in equation (5), so simulation time is getting longer as accumulated number of branch coverage is getting large. The simulation results of SBF reduce 2259 clock cycles with losses of 2 numbers of branch coverage because SBF skips the branch coverage detected in Wts region. The results show that the proposed strategies are more effective.

표 5. 30개의 0 검출 제한을 제거한 SB와 SBF의 비교

Table 5. Comparisons of SB and SBF without restraint of 30 zero coverage.

	SB		SBF	
	CC	BC	CC	BC
S1	1854	128	1460	128
S8	2181	208	1456	208
S11	1982	245	1460	247
S12	2080	364	1462	360
SUM	8097	945	5838	943

V. Conclusion

This paper proposed strategies that could be used to reduce clock cycles in behavioral model verification. The techniques were applied in 12 sample programs and the simulation results showed that a number of clock cycles were reduced with few losses of branch coverage. The work for finding a saturation point of branch coverage increase will be studied in the future.

참 고 문 헌

[1] Janick Bergerson, Writing testbenches : functional verification of HDL model, Kluwer Academic, 2000.

[2] Jim Lipman, "Covering your HDL chip-design bets," EDN, Oct. 22, 1998, pp 65-70.

[3] Brian Barrera, "Code coverage analysis-essential to a safe design," Electronic Engineering, pp 41-43, Nov. 1998.

[4] Kevin Skahill, "A designer's guide to VHDL design and verification," Engineering Software, pp 149-158, Feb. 1996.

[5] A. Von Mayrhauser, et. Al., "On choosing test criteria for behavioral level hardware design verification", IEEE International Hign Level Design Validation and Test Workshop, Berkeley, CA, 2000.

[6] S. Surya, P. Bose, and J. Abraham, "Architectural Performance Verification: PowerPC Processors", IEEE International Conference on Computer Design : VLSI in Computers and Processors, pp. 344-347, Oct. 1994.

[7] J. Gately, "Verifying a million gate processor", Integrated System Design, pp 19-24, 1997.

[8] M. Sahinoglu, A. Von Mayrhauser, A. Hajjar, T. Chen, C. Anderson, "On the efficiency of a compound Poisson stopping rule for mixed strategy testing", IEEE Aerospace conference, Track 7, Mar. 1997.

[9] A. Goel, "Software reliability models : assumptions, limitations, and application", IEEE transactions on software engineering, vSE-11,n 12, pp 1411-1423, Dec. 1985.

[10] W. Howden, "Confidence-based reliability and statistical coverage estimation", Proceedings on the international symposium on software reliability engineering, pp 283-291, Nov. 1997.

[11] S. Chen, S. Mills, "A binary markov process model for random testing", IEEE transactions on software engineering, v22, n3, pp 218-223, Mar. 1996.

[12] Amjad Fuad A. Hajjar, "Bayesian Based Stooping Rules for Behavioral VHDL Verification", Ph. D Dissertation, Fall 2000.

[13] R. Ramchandani, D. Thomas, "Behavioral test generation using mixed integer nonlinear programming", International test conference, pp 958-967, 1994.

[14] G. Hayek, C. Robach, "From specification validation to hardware testing : a unified method", International Test conference, pp 885-893, 1996.

[15] T. Chen, M. Sahinoglu, A. Mayrhauser, A. Hajjar, A. Anderson, "Achieving the quality for behavioral models with minimum effort", 1st international symposium on quality in electronic design, pp , Mar. 2000.

저 자 소 개

金 强 哲(正會員) 第38卷 SD編 第1號 參照