行政院國家科學委員會專題研究計畫　成果報告

即時系統全符號式驗證工具的底層基礎技術(3/3)

計畫類別： 個別型計畫
計畫編號： NSC92-2213-E-002-103-
執行期間： 92 年 08 月 01 日至 93 年 07 月 31 日
執行單位： 國立臺灣大學電機工程學系暨研究所

計畫主持人： 王凡

報告類型： 完整報告

處理方式： 本計畫可公開查詢

中　華　民　國 93 年 12 月 15 日

行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※　即時系統全符號式驗證工具的底層基礎技術（3/3）※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：■個別型計畫　　□整合型計畫

計畫編號：NSC　92－2213－E－002－103－

執行期間：92 年 8 月 1 日至 93 年 7 月 31 日

計畫主持人：王　凡

共同主持人：

執行單位：國立臺灣大學電機工程學系暨研究所

中　華　民　國　　93 年　　5 月　　30 日

行政院國家科學委員會專題研究計畫成果報告

即時系統全符號式驗證工具的底層基礎技術（3/3）

Fundamental Technologies for Verification Tool of Real-Time System

計畫編號：NSC 92-2213-E-002-103
執行期限：92 年 8 月 1 日至 93 年 7 月 31 日

主持人：王　凡　副教授
國立臺灣大學 電機工程學系暨研究所

一、中文摘要

感謝國科會三年的長期支持，我們更深入發展能將數學模型、問題分析、邏輯驗證等相關技術加以結合，而開發出的軟體工具 RED。除了利用功能強大的資料結構有效節省許多空間與時間，並可自動進行高效率複雜即時系統驗證之外，對其底層的基礎技術，更加入了涵蓋率的應用，以預測執行效能，找出合適乃至最佳化的驗證方式，向自動化驗證邁進了一大步，亦提高在業界實務上的應用價值。

關鍵詞：即時系統、驗證、涵蓋率、模型檢驗、全符號式模擬

Abstract

Thanks to the support of National Science Council for the long-term support of three years, we were able to develop the fundamental technologies of the verification tool, RED (Region-Encoding Diagram). By using new data-structures, analysis techniques, and verification techniques, this tool can carry out efficient state-space representation and manipulations. Besides, we add the application of coverage estimation in the tool to predict the performance and efficiency of strategies. According the data , we can find the best or the adaptive methods to execute. We improve the applicability and value of RED in industry.

Keywords: Real-time system, verification , coverage, model checking, symbolic simulation

二、緣由與目的

Presently, with verification and integration costs increasing to more than 50 percent of the development budget in real-world projects, it is more and more difficult to use traditional simulation technology to acquire enough trace coverage to confidently create system designs. As well, application of the new formal verification technology is still hampered by its intrinsic complexity. In the foreseeable future, we expect that simulation and formal verification will be combined for verification of large-scale real-time systems. Symbolic simulation is such a combination . It uses symbolic techniques of formal verification to represent space of simulation traces so that abstract (as opossed to concrete) behaviors can be observed in a trace.

Current simulation technology for real-time systems is still not as developed as that for untimed systems like Very Large Scale Integration (VLSI) Systems. For one thing, the important concept of *coverage* can be used to both estimate the value of a set of traces and to direct the generation of new traces. In short, coverage is how much has

been verified of the target to be verified. The importance of this concept is that, in real-world projects, it is usually the case that we lake enough resources to either run enough traces to obtain confidence, or to complete formal verification tasks. Product designs usually need to be released before we can have 100% confidence in the designs. Therefore it is important that we have some types of coverage metrics to evaluate confidence in our designs

三、研究方法及成果

A good coverage estimation should tell us what proportion of a target function has been covered.

After experimenting with various coverage metrics and their computation methods, we identified the following four criteria for effective numerical coverage metrics.

◦ **accountability:** This ensures that each portion of the target function is accounted for once and only once. If accountability is not maintained, we may run into two bizarre situations. First, some portions may not be accounted for and thus engineers simply cannot trust the metrics to check if all function portions have been covered. Second, it may happen that some portions are counted more than once and thus full coverage estimation is greater than 100% which makes no sense at all. Thus, accountability is the most important criterion.

• **coverability:** This means that $\_p \in V \_(p) = \_p \in F \_(p)$ can be expected at the end of a symbolic simulation if enough traces have been generated. This is desirable in that 100% coverage can be the goal for verification. Moreover, if engineers decide to stop verification at 80% coverage, they can roughly estimate confidence in their products.

• **efficiency:** This criterion measures the overhead in the computation of both the *f* and the *v* in procedure Symbolic Simulate(). If complex formal reachability analysis is used to compute these two estimations, it is not worthwhile to estimate the coverage. In this work, we base our coverage estimation on transition-countings and state-space abstraction techniques and can efficiently calculate estimations in our three metrics.

• **discernment:** This criterion assesses the capability of a metric to discern risk states. This can be an issue when, in some metrics, risk states and nonrisk states are likely to fall in the same *portion*. A metric that frequently fails to detect existing risk states at a high numerical coverage may give users unjustified and false confidence on their system designs.

We use three methods of coverage metric:

1. TA arc coverage metric (ACM)

This is a straightforward adaptation from the technology of VLSI simulation and testing. In the computation of FSM arc coverage for VLSI, we conceptually transform a circuit to a finite-state automaton (FSM) and use the set of alreadytriggered transitions as *V* and the set of executable transitions as *F* to compute coverage estimation. The same definition of FSM arc coverage can be readily copied for the simulation of timed automata (TA). That is, we can also use the arcs of TAs to estimate coverage in the *TA arc coverage metric (ACM)*.Each portion corresponds to an executable transition and the estimation function *_ACM*() maps every portion to 1. The numerical estimation *f* of the whole targetfunction of procedure Symbolic Simulate() can be $|T|$, where *T* is the set of transitions in the TA.

Lemma 1. *ACM for dense-time systems satisfies the accountability criterion.*

Proof : It is true since $\_ACM(e) = 1$ for

every executable transition in F, and we directly use the sizes of already-triggered and executable transition sets to calcualte the coverage. _ The criterion of full coverability is not guaranteed. But as can be seen from our experiment data in section 8, with a tight estimation of the set of executable transitions, it is possible to get very close to 100% coverage. As for the criterion of efficiency, in each iteration, the overhead is a set-union operation and a size calculation of a set with a high efficiency rate. Finally, ACM may not have much discernment since a transition can very often be used in both a safe trace and a trace that ends in a risk state. This means ACM may reach 100% coverage without discovering the risk state even if it exists.

2. Back-and-forth region coverage metric (RCM)

ACM can very often be too coarse to discern risk states. Another extreme that can also be adapted from VLSI verification technology is the *visited-state* coverage metric, which uses the reachable state set in FSM to estimate coverage. The challenge to incorporate the concept into our framework arises from the fact that in VLSI's model, the states are discrete and countable while in timed automata, the states are dense and uncountable. A solution is to use equivalence classes in the dense-time state-space as portions. An equivalence relation to partition dense-time state-space is the *region*-equivalence relation between states. timing constant used in A and the safety state predicate . In this way, we consider in this section the concept of *region coverage metric (RCM)*, in which a basic portion is a region, for the simulation of real-time systems. This coverage metric can have extra leverage with symbolic simulation since, in one step, we may generate a huge proportion of the state-space represented by a set of symbolic constraints.RCM has enough discerning power to discover reachable risk states whereas ACM lacks such discerning power.

3. Triggering-condition coverage metric (TCM)

RCM has the advantage in accountability and discernment. But it may result in low coverability since our estimation of the reachable region sets can be imprecise.

On the other hand, ACMcan suffer from low discernment. In this section, we propose a balanced approach called *triggering-condition coverage metric (TCM)*, in which we use the triggering conditions of all transitions as the body of the whole target function. TCM estimates the proportion of the covered triggering conditions of all transitions. It is accounted as the summation of triggering condition coverage for each transition. Formally speaking, a basic portion in TCM is a pair like (e, γ) where e is an executable transition and γ is a region in subspace $\tau(e)$ (the triggering condition of e). The numerical estimation f of the whole target function in procedure.

4.Experimental Result

iteration	ACM	ACM time overhead	RCM	RCM time overhead	TCM	TCM time overhead
1	4/97	0.00sec.	0.167816	7.39sec.	0.004092	0.02sec.
2	8/97	0.00sec.	0.173442	7.39sec.	0.382901	0.02sec.
3	12/97	0.00sec.	0.174279	7.40sec.	0.783131	0.02sec.
4	20/97	0.01sec.	0.175273	7.41sec.	0.799498	0.02sec.
5	36/97	0.02sec.	0.232154	7.41sec.	0.813138	0.03sec.
6	42/97	0.03sec.	0.295386	7.41sec.	0.815525	0.04sec.
7	64/97	0.05sec.	0.408160	7.42sec.	0.884971	0.06sec.
8	76/97	0.08sec.	0.561395	7.43sec.	0.920890	0.08sec.
9	88/97	0.12sec.	0.956820	7.44sec.	0.971241	0.11sec.
10	94/97	0.17sec.	0.965724	7.45sec.	0.975507	0.15sec.
11	94/97	0.22sec.	0.974428	7.46sec.	0.975507	0.18sec.
12	95/97	0.28sec.	0.975538	7.48sec.	0.976530	0.22sec.
13	97/97	0.34sec.	0.975783	7.49sec.	1.000000	0.26sec.
14	97/97	0.40sec.	0.981319	7.50sec.	1.000000	0.29sec.
15	97/97	0.47sec.	0.981338	7.52sec.	1.000000	0.33sec.
16	97/97	0.55sec.	0.982733	7.54sec.	1.000000	0.36sec.
17	97/97	0.63sec.	0.982734	7.56sec.	1.000000	0.40sec.
18	97/97	0.70sec.	0.982734	7.57sec.	1.000000	0.44sec.

Strategy	Faulty Models	Depth	ACM	RCM	TCM	Risk state reached?
Depth First	1	66	71/98	0.355262	0.958950	Yes
	2	64	71/98	0.354993	0.958950	Yes
	3	26	27/97	0.293884	0.437878	Yes
	4	96	90/97	0.924978	0.963982	Yes
	5	64	63/97	0.355688	0.948209	Yes
	6	62	61/98	0.357885	0.936412	Yes
Breath First	1	11	95/98	0.932724	0.975532	Yes
	2	11	95/98	0.943274	0.975532	Yes
	3	9	88/97	0.564228	0.971241	Yes
	4	7	64/97	0.294859	0.884971	Yes
	5	10	92/97	0.898077	0.973172	Yes
	6	9	79/98	0.660754	0.949243	Yes

四、結論與討論

Symbolic simulation combines the advantages of both simulation and formal verification and can be an important verification approach before fully automatic formal verification becomes applicable. In this paper, we present techniques for coverage estimation for dense-time systems. We hope such techniques can be the solid stepstone toward the development of powerful symbolic simulators for industry real-time systems. Many issues raised in this work also deserve future research. For example, it will be interesting to see the design of quantitative metrics for our criterion of discernment in the symbolic simulation of dense-time systems. With such metrics, the criterion becomes equivalent to the notion of observability.

五、參考文獻

1. R. Alur, C. Courcoubetis, D.L. Dill. Model Checking for Real-Time Systems, IEEE LICS, 1990.
2. R. Alur, D.L. Dill. Automata for modelling real-time systems. ICALP' 1990, LNCS 443, Springer-Verlag, pp. 322-335.
3. J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, L.J. Hwang. Symbolic Model Checking: 1020 States and Beyond, IEEE LICS, 1990.
4. B.Beizer. Software Testing Techniques. Van Nostrand Rheinhold, New York, 2nd ed., 1990.
5. L. Bening, H. Foster. Principles of Verifiable RTL Design, a Functional Coding Style Supporting Verification Processes in Verilog, 2nd ed., Kluwer Academic Publishers,2001.
6. G. Behrmann, K.G. Larsen, J. Pearson, C. Weise, Wang Yi. Efficient Timed Reachability Analysis Using Clock Difference Diagrams. CAV'99, July, Trento, Italy,LNCS 1633, Springer-Verlag.
7. R.E. Bryant. Graph-based Algorithms for Boolean Function Manipulation, IEEE Trans. Comput., C-35(8), 1986.
8. K-T. Cheng and A. S. Krishnakumar. Automatic Functional Test Generation Using the Extended Finite State Machine Model. In Proceedings of the 30th Design Automation Conference, pp. 86-91, June 1993.
9. Hana Chockler, Orna Kupferman, Moshe Y. Vardi. Coverage Metrics for Temporal Logic Model Checking. LNCS 2031, pp. 528-552, 2001.
10. S. Devadas, A. Ghosh, K. Keutzer. An Obervability-Based Code Coverage Metric f or Functional Simulation. IEEE ICCAD'96.
11. D.L. Dill. Timing Assumptions and Verification of Finite-state Concurrent Systems.CAV'89, LNCS 407, Springer-Verlag.
12. D.L. Dill. What's Between Simulation and Formal Verification? In Proceedings of DAC1998.
13. Farzan Fallah, Srinivas Devadas, Kurt Keutzer. OCCOM: Efficient Computation of Observability-Based Code Coverage Metrics for Functional Verification. Design Automation Conference, pp. 152-157, 1998.
14. D. Geist, M. Farkas, A. Landver, Y.Lichtenstein, S. Ur, Y. Wolsfthal. Coverage-Directed Test Generation Using Symbolic Techniques. In Proceedings of the Int'l Conference on Formal Methods in CAD, November 1996.
15. J. Haartsen. Bluetooth Specification, version 1.0. http://www.bluetooth.com/
16. T.A. Henzinger, X. Nicollin, J. Sifakis, S. Yovine. Symbolic Model Checking for Real-Time Systems, IEEE LICS 1992.
17. C.A.R. Hoare. Communicating Sequential Processes, Prentice Hall, 1985.
18. Y. Hoskote, D.Moundanos, J. Abraham. Automatic Extraction of the Control Flow Machine and Application to Evaluating

Coverage of Verification Vectors. Proceedings of ICCD, pp. 532-537, October 1995.
19. Y. Hoskote, D. Moundanos, and J. A. Abraham. Automatic Extraction of the control flow machine and application to evaluating coverage of verification vectors.
In Proceedings of the Int'l Conference on Computer Design, pp 532-537, October 1995.
20. Y. Hoskote, T. Kam, P.-H. Ho, and X. Zhao. Coverage estimation for symbolic model checking. In Proceedings of DAC1999, pp. 300-305.
21. Kim G. Larsen, Paul Pettersson and Wang Yi. Model-Checking for Real-Time Systems. In Proceedings of the 10th International Conference on Fundamentals of Computation Theory, Dresden, Germany, August 22-25, 1995. LNCS 965, pp.62-88, Horst Reichel.
22. B. Mike, G. Daniel, H. Alan, W. Yaron, M. Gerard, S. Ralph. A Study in Coverage-
Driven Test Generation. In Proceedings of DAC1999, pp. 970-975.
23. P. Pettersson, K.G. Larsen. UPPAAL2k. In Bulletin of the European Association for Theoretical Computer Science, Vol. 70, pp. 40-44, 2000.
24. P. Rashinkar, P. Paterson, L. Singh. System-on-a-chip Verification, Methodology and Techniques. Kluwer Academic Publishers, 2001.
25. C.-J.H. Seger, R.E. Bryant. Formal Verification by Symbolic Evaluation of Partially-Ordered Trajectories. Formal Methods in System Designs, Vol. 6, No. 2, pp. 147-189, Mar. 1995.
26. F. Wang. Efficient Data-Structure for Fully Symbolic Verification of Real-Time Software Systems. TACAS'2000, March, Berlin, Germany. LNCS 1785, Springer-Verlag.
27. F. Wang. Symbolic Verification of Complex Real-Time Systems with Clock-Restriction Diagram, to appear in Proceedings of FORTE, August 2001, Cheju Island, Korea.
28. F.Wang. Efficient Verification of Timed Automata with BDD-like Data-Structures. to appear in proceedings of VMCAI 2003, New York City, USA, Jan. 2003. LNCS 2575, Springer-Verlag.
29. F. Wang, G-D. Hwang, F. Yu. Symbolic Simulation of Real-Time Concurrent Systems to appear in proceedings of RTCSA 2003, Tainan, Taiwan. Feb. 2003. LNCS,Springer-Verlag;
30. S. Yovine. Kronos:A Verification Tool for Real-Time Systems. International Journal of Software Tools for Technology Transfer, Vol. 1, Nr. 1/2, October 1997.