International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 2, February - 2014

A Review on Multicore Architecture and their Validation

Ramalakshmi. R,
M. E. Embedded system technologies,
Sri Shakthi Institute of Engineering and Technology,

Rajamunipriya. R,
M. E. Embedded system technologies,
Sri Shakthi Institute of Engineering and Technology,

***Abstract*— Multicore processors are widely used in today's servers, desktop and embedded systems. It is a major challenge to verify functional correctness as well as non-functional requirements of multicore architectures. Direct application of existing functional validation approaches usually consumes too much time to reach the coverage goal due to the complexity of multicore designs. Escaped bugs can lead to serious consequences in many scenarios. Due to parallel execution of task sets, existing approaches are also insufficient to validate whether applications in such systems can be scheduled within the given temperature, energy, and timing constraints. If these constraints are violated, it can lead to performance degradation or even catastrophic consequences in safety-critical systems.**

I. INTRODUCTION

Multicore architectures are widely used in todays desktop, server, and embedded systems. Due to the existence of power wall, conventional single core architectures can no longer deliver the required performance improvement by increasing frequency. Instead, architects integrate more and more cores into the same chip to boost the throughput. By operating multiple cores at a lower frequency, multicore architectures can achieve the same performance with much smaller power dissipation compared with a high clock rate monolithic core. For desktop-based systems and servers, the multicore architectures ensure the fast growth of computation performance along with time. With rapid adoption of dual-core and quad-core processors, we are moving down the path to 32, 64 or even hundreds of cores. For embedded systems, the energy efficiency introduced by multicore architectures allows devices to operate for longer time with the same battery capacity. Besides, since multiple cores are sharing the same die, the Printed Circuit Board (PCB) size is also reduced.

With the growing demand for green data-centers, long-life computers and handhold devices, multicore architectures will continue to dominate the design of next generation System-on-Chip (SoC) architectures. Successful multicore designs must satisfy both functional and non-functional requirements. Functional requirements ensure that the processor performs all logical functions as specified by the design specification. Non-functional requirements are imposed to make the design satisfy various design constraints such as area, power, energy, temperature, and performance. Clearly, functional requirements are important, because a buggy (erroneous) design leads to unreliable systems. Depending on application domains, unreliable systems can cause loss of vital information or even disaster. Non-functional requirements are also equally important, because violation of non-functional requirements can also lead to serious consequences. For example, due to uneven activities on different cores, the die temperature of busy cores can easily reach 120 C [16]. If the high die temperature is not well controlled, the transient error occurs more frequently and the device is less reliable. Also, devices that always operate in high temperature usually have much shorter lifespan as shown in industrial studies [82]. To avoid these unwanted scenarios, both functional and non-functional validation must be performed to ensure the success of modern multicore designs.

1.1 Functional Validation Of Multicore Architectures

While multicore architectures are very successful to boost the throughput, their increasing complexity also introduces significant validation challenges. Most widely used functional validation techniques are based on simulation using random and constrained-random tests [93] [1] [83]. The multicore design is placed within a simulation environment and a test generator randomly feeds new tests into the design. The behaviour of the design under test is compared with the golden reference model to detect any functional errors. As illustrated in Figure 1-1 [77], the verification complexity grows tremendously in last two decades. Due to the increasing complexity of multicore architectures, even trillions of simulation vectors may not be inadequate to achieve the required coverage goal within ever decreasing time-to-market window.

Since simulation vectors are generated randomly, it is quite difficult for random tests to activate coverage holes. Directed tests [22] are promising to address this problem. By analyzing the logical structure of the design, a small number of directed tests can activate the desired behavior of the system. They can be applied in addition to the random tests to reach the coverage goal with much less time. Unfortunately, most directed tests are manually

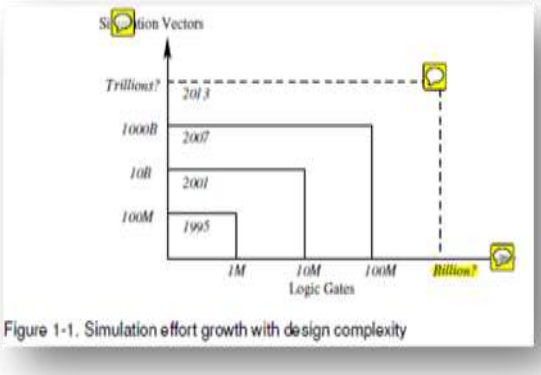

Figure 1-1. Simulation effort growth with design complexity

written, which is time consuming and error-prone. Fully automatic directed test generation schemes are desired to accelerate the verification process of multicore architectures. There are two major objectives in directed test generation. First, the overall validation effort should be minimized by reducing the total number of tests required to achieve the coverage goal. Secondly, test generation time should also be small.

Model checking [13, 28] is promising for automatic test generation. To activate a particular scenario, we can feed the negated version of a property to the model checker, and use the resultant counterexample as a directed test. Due to the state space explosion problem, such a process is usually quite time consuming. Since different cores in a multicore design usually contain similar structure, their formal descriptions (such as CNF in SAT-based model checking) also exhibits significant symmetry. We believe such symmetry can be exploited to accelerate the model checking process, because the information we learn from one core may be applied directly to other cores. Unfortunately, this intuitive reasoning is hard to implement because it is very difficult to reconstruct the symmetry from the CNF formula. The high level information is lost during CNF synthesis, and it is inefficient as well as computationally expensive to recover through "reverse engineering" methods.

An important requirement of functional validation is to achieve certain state or transition coverage metric in the state space of the design. Random simulation is widelyused in industry to fulfill this goal. However, due to the symmetric nature of multicore architecture, its state space contains some unique features, which can be utilized to reduce the test length or testing time required to reach the required coverage goal. Although the FSM of each cache controller is easy to understand, the structure of the product FSM for modern cache coherence protocols usually have quite obscure structures that are hard to analyze. Besides, modern processors usually contain multiple cache levels, which greatly complicates the global state space. Even if the global state space can be described, it is still difficult to find an efficient way to perform traversal in it. In other words, the test generation algorithm must activate all states and transitions with limited number of unnecessary transitions. Moreover, since the state space is quite large, the tests usually introduce a large storage overhead. Therefore, it is desirable that the test can be generated on the fly.

1.2 Validation of Non-functional Requirements

So far we have described the importance of ensuring functional correctness and challenges associated with verifying multicore architectures. It is also equally important to ensure that all the non-functional requirements are met. One of the key problems is to find whether a given task set can be scheduled on the processor(s) without violating the required temperature and energy constraints. This kind of validation is important to ensure the reliability of multicore designs, because high die temperature leads to more frequent transient errors as well as shorter processor lifespan [82].

Besides, the management of overall energy consumption is also crucial to the success of an embedded design. Since many handheld devices are equipped with multicore processors but still battery-powered, we need to validate that all important tasks are finished with limited energy consumption. It is usually very costly to perform such validation, because the manufacturer need to build the full system and test the design by executing real task set. Since the worst case behavior of real-time systems usually can be obtained by offline analysis, we believe it is possible to predict the system behavior based on the information collected via static analysis of task sets and execution environment. In other words, a large portion of non-functional validation can be performed without running the actual system in real environments. The major challenge in this field comes from the NP-hard nature [103] [100] [86] of the schedulability problem. In fact, it is NP-hard even to verify the schedulability of a task set under temperature and energy constraints in a single core processor. The problem is more complex when the system contains multiple cores.

2. SYSTEM-LEVEL VALIDATION TECHNIQUES

For ease of presentation, we have divided the existing approaches into three categories. First, we describe the test generation approaches for architecture validation. Next, we discuss existing techniques for cache coherence protocols validation. Finally, we present techniques for validation of non-functional requirements.

2.1. Test Generation for Architecture Validation

Model checking techniques are promising for functional verification and test generation of complex systems [39, 50, 51, 64]. Figure 2-1 shows the general

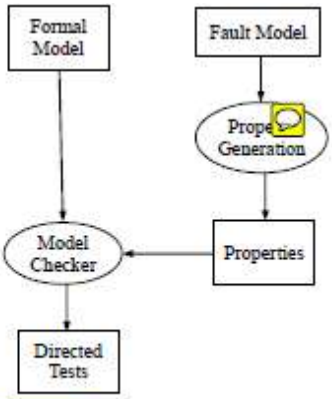

Figure 2-1. Directed test generation

framework of the directed test generation based on model checking. In order to create directed tests, the formal model of the design specification and a suitable fault model are provided as input. Then a set of properties are generated for the desired behaviours (faults) that should be activated in the simulation based validation stage. For example, when a graph model of the design and a functional coverage fault model is provided, a coverage-driven property generation can be used. For circuits with stuck-at fault model, the property will be in the form of G(a = 1) or G(a = 0). After that, a model checker is employed to check wether there exists some states which violate the negated version of the property. It reports a counterexample, if it finds a violation. This counter example contains a sequence of input information which will drive the system from an initial state to a state, which does not satisfy the negated version of the property, or in other words, which satisfies the original property. Therefore, we can use it as a test to activate the corresponding property or behavior during simulation-based validation. Although model checking is effective for directed test generation, the capacity of the conventional symbolic model checking is usually limited. Bounded model checking (BMC) was proposed to address this problem by checking whether there is a counterexample for the property within a given bound [13] [28]. BMC cannot prove the validity of a safety property to hold globally when no Counter example is found within a specific bound, but it is quite effective to falsify a design when the bound is not large. The reason is that SAT solvers usually require less space and time than conventional Binary Decision Diagram (BDD) based model checkers [65]. Therefore, SAT-based BMC is suitable for directed test generation [64], where a counterexample typically exists within a relatively small bound. To generate the directed test, the negated version of the property is checked by BMC. The SAT solver will find an assignment of all input and state variables, which satisfies (2–1). As a result, we can extract the assignment sequence of input variables and use it as a test to activate the desired property in the system.

A great deal of work has been done to reduce the SAT solving time during BMC [22–25, 43, 52, 79, 91]. The basic idea is to exploit the regularity of the SAT instances between different bounds. For example, incremental SAT solvers [43, 91] reduce the solving time by employing the previously learned conflict clauses. Generated conflict clauses are kept in the database as long as the clauses which led to the conflicts are not removed. Strichman [79] proposed that if a conflict clause is deduced only from the transition part of a SAT instance, it can be safely forwarded to all instances with larger bounds, because the transition part of the design will still be in the SAT instance when we unroll the design for more times. Besides, the learned conflict clauses can also be replicated across different time steps. However, the existing approaches did not exploit the symmetric structure within the same time step. In directed test generation for multicore architectures, same knowledge about the core structure needs to be re-discovered for each core independently, which can lead to significant wastage of computational power. When BMC is applied in circuits, Kuehlmann [53] proposed that the unfolded transition relation can be simplified by merging vertices that are functionally equivalent under given input constraints. In this way, the complexity of transition relation is greatly reduced. Since this technique is based on the AIG representation of logic designs, it is difficult to use for accelerating the solving process of CNF instances, which are directly created from high level specifications. Verification and validation based on high level specification are proved to be effective. For example, Bhadra et al. [45] used executable specification to validate multiprocessor systems-on-chip designs. Chen et al. [22] proposed directed test generation based on high level specification. To accelerate the test generation process, conflict clauses learned during checking of one property are forwarded to speed up the SAT solving process of other related properties, although the bound is required as an input. Similarly, the simultaneous SAT solver [49] enabled the learned clauses to be reused by properties. Decision ordering was also studied in [23] to reduce the SAT solving time. These approaches did not take the advantage of structural symmetry in multicore architectures.

When SAT instance contains symmetric structure, symmetry breaking predicate [3, 5, 30, 62, 80] can be used to speed up the SAT solving by confining the search to non-symmetric regions of the space. By adding symmetry breaking predicates to the SAT instance, the SAT solver is restricted to find the satisfying assignments of only one representative member in a symmetric set. However, this approach cannot effectively accelerate the directed test generation for multicore processors, because the properties for test generation are usually not symmetric with respect to each core. Thus, the symmetric regions in the entire space are usually small despite the fact that the structure of each core is identical. On the other hand, in component analysis for SAT solving, Biere et al. [14] proposed that each component can be solved individually to accelerate the solving process. However, the symmetric structure is not used at the same time for further speedup.

During the validation process, it is also very important to generate assertions effectively. One important work in this direction is GoldMine [81], which automatically uses data mining and formal verification to generate assertions for real hardware designs. Using the simulation trace of RTL designs, GoldMine employs decision tree based supervised learning algorithms to mine potential assertions from the simulation data. Liu et al. [54] also proposed a methodology, which utilizes GoldMine to achieve coverage closure during design validation. Once the assertion is generated, automatic test generation approaches can be employed to generate the tests, which can be used to activate the desired behavior of the system. For example, test generation tools based on interleaved concrete and symbolic execution, such as DART [40], CUTE [72], and Apollo [7], are promising in capturing important bugs in large software systems. STAR [55] and HYBRO [56] are proposed to generate tests by combining static and dynamic analysis for hardware validation. Due to the effective utilization of the CFG, HYBRO [56] demonstrated remarkable improvement over previous path-

based test generation technique [55]. However, HYBRO cannot be applied on designs containing dynamic array references.

2.2 Validation of Cache Coherence Protocols

Verification of cache coherence protocols for multicore and multiprocessor systems has been widely studied in both academia and industry. Existing studies can be broadly grouped into two categories: formal verification [27, 33, 36] and simulation based validation [2, 83, 93]. Formal methods using model checking can prove mathematically whether the description of certain cache coherence protocol violates the required property. For example, Mur' [33] was designed and used to verify various cache coherence protocols based on explicit model checking. Counter-example guided refinement [27] is employed to verify complex protocols with multilevel caches. Besides, symbolic model checking tools are also developed for coherence verification. For example, Emerson et al. [36] investigated the verification problem with parameterized cache coherence protocol using BDDs. Although formal methods can guarantee the correctness of a design, they usually require that the design should be described in certain input languages. As a result, model checking usually cannot be applied to implementations directly Simulation based approaches, on the other hand, are able to handle designs at different abstraction levels and therefore more widely used in practice. For example, Wood et al. [93] used random tests to verify the memory subsystem of SPUR machine.

Successive loads and stores to the same location are employed as test template to expose possible errors. Genesys Pro test generator [2] from IBM extended this direction with more complex and sophisticated test templates. To reduce the search space, Abts et al. [1] introduced space pruning technique during their verification of the Cray processor. Wagner et al. [83] designed the MCjammer tool which can get higher state coverage than normal constrained random tests. Existing random test generation tools are proven to be effective to discover potential bugs. However, due to their random nature, it is very hard to achieve full state and transition coverage in a reasonable time. Since an uncovered transition can only be visited by taking a unique action at a particular state, it may not be feasible for a random test generator to eventually cover all possible states and transitions. To address this problem, some random testers are equipped with small amount of memory, so that the future search can be guided to the uncovered regions. Unfortunately, unless the memory is large enough to hold the entire state space, it is still quite hard to achieve full coverage by such guided random testing.

2.3 Task Schedulability under Constraints

Energy-aware scheduling techniques for real-time systems have been widely studied to reduce energy consumption. While several works employed dynamic cache reconfiguration [87] [85], most of them are based on Dynamic Voltage Scaling (DVS). Aydin et al. [9] addressed both static and dynamic slack allocation problems for periodic task sets, while Shin et al. [73] also considered aperiodic tasks. Jejurikar et al. focused on energy-aware scheduling for non-preemptive task sets [47] and leakage power minimization [48]. Zhong et al. [103] solved a system-wide energy minimization problem with consideration of other components. Wang et al. [85] proposed a leakage-aware energy saving technique based on DVS as well as cache reconfiguration. As shown in [100], applying DVS in real-time systems is a NP-hard problem. Optimal and approximation algorithms are given in [103] [100] [86], while other works proposed heuristics. A survey on recent works can be found in [21]. However, these techniques are not aware of controlling the operating temperature. Temperature-aware scheduling in real-time systems has drawn significant research interests in recent years. Wang et al. [84] introduced a simple reactive DVS scheme aiming at meeting task timing constraints and maintaining processor safe temperature. Zhang et al. [101] proved the NP-hardness of temperature-constrained performance optimization problem in real-time systems and proposed an approximation algorithm.

Yuan et al. [97] considered both temperature and leakage power impact in DVS problem for soft real-time systems. Chen et al. [20] explored temperature-aware scheduling for periodic tasks in both uniprocessor and homogeneous multiprocessor DVS-enabled platforms. Liu et al. [57] proposed a design-time thermal optimization framework which is able to solve problem variants EA, TA and TCEA scheduling in embedded system with task timing constraints. Jayaseelan et al. [46] exploited different task execution orders, in which each task has distinct power profile, to minimize peak temperature. However, none of these techniques solves TCEC problem. Moreover, they all make certain assumptions on system characteristics that limits their applicability. Existing research formulated the voltage/frequency assignment problems in different models. For example, Integer Linear Programming (ILP) has been widely applied to many voltage/frequency assignment problems without the temperature constraint [94, 102]. Chantem et al. [19] also used ILP to model scheduling problem with steady-state temperature constraints. Unfortunately, when transient temperature is considered, the full expansion of the temperature constraint introduces a large number of product terms, which prevent us to solve the problem efficiently using ILP solvers. Coskun et al. [29] circumvented this problem using an iterative ILP and thermal simulation approach, although the convergence to the optimal solution is not guaranteed.Another important modeling technique is timed automata [6]. Norstorm et al. [66] first extended timed automata with the notion of real-time tasks and showed that the traditional schedulability analysis can be transformed to a decidable reachability problem in timed automata, which can be solved using model checking tools. Fersman et al. [37] further generalized this approach with asynchronous processes and preemptive tasks in continuous-time model. However, none of these techniques considered energy or temperature related issues.

There are several studies on Dynamic Power Management (DPM) using formal verification methods for embedded systems [74] and multiprocessor platforms [58]. Shukla et al. [74] provided a preliminary study on evaluating DPM schemes using an off-the-shelf model checker. Lungo et al. [58] tried to incorporate verification of DPM schemes in the early design stage. They showed that tradeoffs can be made between design quality and verification efforts. None of these approaches considers temperature management in such systems. Moreover, they did not account for energy and timing constraints, which are important in real-time embedded systems. Wang et al. [88] discussed the application of time automata in schedulability problem with both energy and temperature constraints. Nevertheless, due to the capacity limit of model checker, the proposed technique can only be applied to small task sets.

Temperature- or energy-constrained scheduling problems are also related to the multi-constrained path (MCP) problem for Quality of Service (QoS). MCP was extensively studied by network community. For example, Chen et al. [26] designed an approximation algorithm for MCP with two constraints. [76] and [98] studied the efficient heuristics for MCP problems. Xue et al. [96] proposed polynomial time approximation algorithms, which can be applied for more than two constraints. However, since the QoS costs are usually modeled as additive constants, these existing methods cannot be applied directly to solve TCEC problem due to the fact that the computation of the temperature is not additive.

3. CONCLUSION

To design reliable multicore systems, it is crucial to satisfy both functional and non-functional requirements. The functional requirements ensure that the design performs all the logical operations as specified. The non-functional requirements guarantee that the system does not violate various design constraints such as area, power, energy, and temperature. While the complexity of modern multicore architectures are increasing rapidly, it introduces various challenges during validation of both functional behavior and non-functional requirements. In conclusion, this paper presented a comprehensive study of the system-level validation of multicore architectures.

REFERENCES

[1] D. Abts, S. Scott, and D. Lilja. So many states, so little time: verifying memory coherence in the Cray X1. In Proceedings of International Parallel and Distributed Processing Symposium,, 2003.

[2] A. Adir, E. Almog, L. Fournier, E. Marcus, M. Rimon, M. Vinov, and A. Ziv. Genesys-pro: innovations in test program generation for functional processor verification. IEEE Design Test of Computers, 21(2):84 – 93, 2004.

[3] F. A. Aloul, I. L. Markov, and K. Sakallah. Shatter: efficient symmetry-breaking for boolean satisfiability. In Proceedings of Design Automation Conference, pages 836–839, 2003.

[4] F. A. Aloul, I. L. Markov, and K. A. Sakallah. Shatter. University of Michigan, 2003. Available from: http://www.aloul.net/Tools/shatter/.

[5] F. A. Aloul, A. Ramani, I. L. Markov, and K. Sakallah. Solving difficult SAT instances in the presence of symmetry. In Proceedings of Design Automation Conference, pages 731–736, 2002.

[6] R. Alur and D. L. Dill. A theory of timed automata. Theoretical Computer Science, 126(2):183–235, 1994.

[7] S. Artzi, A. Kiezun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. Ernst. Finding bugs in web applications using dynamic test generation and explicit-state model checking. IEEE Transactions on Software Engineering, 36(4):474 –494, 2010.

[8] H. Aydin, R. Melhem, D. Mosse, and P. Mejia-Alvarez. Determining optimal processor speeds for periodic real-time tasks with different power characteristics. In Proceedings of Euromicro Conference on Real-Time Systems, pages 225–232, 2001.

[9] H. Aydin, R. Melhem, D. Mosse, and P. Mejia-Alvarez. Power-aware scheduling for periodic real-time tasks. IEEE Transactions on Computers, 53(5):584–600, 2004.

[10] M. Berkelaar, K. Eikland, and P. Notebaert. lpsolve. Eindhoven Univeristy of Technology, 2010. Available from: http://lpsolve.sourceforge.net/.

[11] D. Bernstein, D. Cohen, and D. Maydan. Dynamic memory disambiguation for array references. In Proceedings of International Symposium on Microarchitecture, pages 105 – 111, 1994.

[12] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu. Bounded model checking. Advances in Computers, 58:118–149, 2003.

[13] A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu. Symbolic model checking without BDDs. In Proceedings of International Conference on Tools and Algorithms for Construction and Analysis of Systems, pages 193–207, 1999. 159

[14] A. Biere and C. Sinz. Decomposing SAT problems into connected components. Journal on Satisfiability, Boolean Modeling and Computation, 2:191–198, 2006.

[15] N. Binkert, R. Dreslinski, L. Hsu, K. Lim, A. Saidi, and S. Reinhardt. The M5 Simulator: Modeling Networked Systems. IEEE Micro, 26(4):52 –60, 2006.

[16] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter variations and impact on circuits and microarchitecture. In Proceedings of Design Automation Conference, pages 338–342, 2003.

[17] R. E. Bryant. Graph-based algorithms for boolean function manipulation. IEEE Transactions on Computers, 35(8):677–691, 1986.

[18] R. Cavada, A. Cimatti, C. A. Jochim, G. Keighren, E. Olivetti, M. Pistore, M. Roveri, and A. Tchaltse. NuSMV. ITC-Irst, 2010. Available from: http://nusmv.irst.itc.it/.

[19] T. Chantem, R. P. Dick, and X. S. Hu. Temperature-aware scheduling and assignment for hard real-time applications on mpsocs. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 288–293, 2008.

[20] J.-J. Chen, C.-M. Hung, and T.-W. Kuo. On the minimization for the instantaneous temperature for periodic real-time tasks. In Proceedings of IEEE Real Time and Embedded Technology and Applications Symposium, pages 236–248, 2007.

[21] J.-J. Chen and C.-F. Kuo. Energy-efficient scheduling for real-time systems on dynamic voltage scaling (dvs) platforms. In Proceedings of IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, pages 28–38, 2007.

[22] M. Chen and P. Mishra. Functional test generation using efficient property clustering and learning techniques. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29(3):396 –404, 2010.

[23] M. Chen and P. Mishra. Decision ordering based property decomposition for functional test generation. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 167–172, 2011.

[24] M. Chen and P. Mishra. Property learning techniques for efficient generation of directed tests. IEEE Transactions on Computers, 60(6):852–864, 2011.

[25] M. Chen, X. Qin, and P. Mishra. Efficient decision ordering techniques for sat-based test generation. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 490–495, 2010.

[26] S. Chen and K. Nahrstedt. On finding multi-constrained paths. In Proceedings of IEEE International Conference on Communications, volume 2, pages 874 –879, 1998. 160

[27] X. Chen, Y. Yang, M. Delisi, G. Gopalakrishnan, and C.-T. Chou. Hierarchical cache coherence protocol verification one level at a time through assume guarantee. In Proceedings of IEEE International High Level Design Validation and Test Workshop, pages 107 –114, 2007.

[28] E. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. Formal Methods in System Design, 19(1):7–34, 2001.

[29] A. Coskun, T. Rosing, K. Whisnant, and K. Gross. Static and dynamic temperature-aware scheduling for multiprocessor socs. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 16(9):1127–1140, 2008.

[30] P. T. Darga, M. H. Liffiton, K. A. Sakallah, and I. L. Markov. Exploiting structure in symmetry detection for cnf. In Proceedings of Design Automation Conference, pages 530–534, 2004.

[31] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. Communication of ACM, 5(7):394–397, 1962.

[32] M. Davis and H. Putnam. A computing procedure for quantification theory. Journal of ACM, 7(3):201–215, 1960.

[33] D. Dill, A. Drexler, A. Hu, and C. Yang. Protocol verification as a hardware design aid. In Proceedings of International Conference on Computer Design, pages 522–525, 1992.

[34] B. Dutertre and L. M. de Moura. A fast linear-arithmetic solver for DPLL(T). In Proceedings of International Conference on Computer Aided Verfication, pages 81–94, 2006.

[35] J. Edmonds and E. L. Johnson. Matching, Euler Tours, and the Chinese Postman. Mathematical Programming, 5:88–124, 1973.

[36] E. Emerson and V. Kahlon. Exact and efficient verification of parameterized cache coherence protocols. In Proceedings of IFIP WG 10.5 Advanced Research Working Conference on Correct Hardware Design and Verification Methods, volume 2860, pages 247–262, 2003.

[37] E. Fersman, P. Pettersson, and W. Yi. Timed automata with asynchronous processes: Schedulability and decidability. In Proceedings of International Conference on Tools and Algorithms for the Construction and Analysis of Systems, pages 67–82, 2002.

[38] Z. Fu, Y. Mahajan, and S. Malik. zChaff. Princeton University, 2001.Availablefrom: http://www.princeton.edu/~chaff/zchaff.html.

[39] A. Gargantini and C. Heitmeyer. Using model checking to generate tests from requirements specifications. In Proceedings of the 7th European Software Engineering Conference held jointly with the 7th ACM SIGSOFT International 161 Symposium on Foundations of Software Engineering, volume 24, pages 146–162, 1999.

[40] P. Godefroid, N. Klarlund, and K. Sen. Dart: directed automated random testing. In Proceedings of the ACM SIGPLAN conference on Programming language design and implementation, pages 213–223, 2005.

[41] M. Guthaus, J. Ringenberg, D.Ernest, T. Austin, T. Mudge, and R. Brown. Mibench: A free, commercially representative embedded benchmark suite. In Proceedings of IEEE International Workshop on Workload Characterization, pages 3–14, 2001.

[42] J. Hennessy and D. Patterson. Computer Architecture: A Quantitative Approach. Morgan Kaufmann Publishers, 2003.

[43] J. N. Hooker. Solving the incremental satisfiability problem. Journal of Logic Programming, 15(1-2):177–186, 1993.

[44] W. Huang, K. Sankaranarayanan, K. Skadron, R. J. Ribando, and M. R. Stan. Accurate, pre-rtl temperature-aware design using a parameterized, geometric thermal model. IEEE Transactions on Computers, 57:1277–1288, 2008.

[45] M. A. J. Bhadra, E. Trofimova. Validating power architecture technology-based mpsocs through executable specifications. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 16(4):388–396, 2008.

[46] R. Jayaseelan and T. Mitra. Temperature aware task sequencing and voltage scaling. In Proceedings of IEEE/ACM International Conference on Computer- Aided Design, pages 618–623, 2008.

[47] R. Jejurikar and R. Gupta. Energy aware non-preemptive scheduling for hard real-time systems. In Proceedings of Euromicro Conference on Real-Time Systems, pages 21–30, 2005.

[48] R. Jejurikar, C. Pereira, and R. K. Gupta. Leakage aware dynamic voltage scaling for real-time embedded systems. In Proceedings of Design Automation Conference, pages 275–280, 2004.

[49] Z. Khasidashvili, A. Nadel, A. Palti, and Z. Hanna. Simultaneous SAT-based model checking of safety properties. In Proceedings of Haifa Verification Conference, pages 56–75, 2005.

[50] H.-M. Koo and P. Mishra. Functional test generation using property decompositions for validation of pipelined processors. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 1240–1245, 2006.

[51] H.-M. Koo and P. Mishra. Test generation using SAT-based bounded model checking for validation of pipelined processor. In Proceedings of ACM Great Lakes Symposium on VLSI, pages 362–365, 2006. 162

[52] H.-M. Koo and P. Mishra. Functional test generation using design and property decomposition techniques. ACM Transactions on Embedded Computing Systems,8(4):32:1–32:33, 2009.

[53] A. Kuehlmann. Dynamic transition relation simplification for bounded property checking. In Proceedings of IEEE/ACM International Conference on Computer- Aided Design, pages 50–57, 2004.

[54] L. Liu, D. Sheridan, W. Tuohy, and S. Vasudevan. Towards coverage closure: Using goldmine assertions for generating design validation stimulus. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 173–178, 2011.

[55] L. Liu and S. Vasudevan. Star: Generating input vectors for design validation by static analysis of RTL. In Proceedings of IEEE HLDVT Workshop, 2009.

[56] L. Liu and S. Vasudevan. Efficient validation input generation in rtl by hybridized source code analysis. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 1–6, 2011.

[57] Y. Liu, H. Yang, R. P. Dick, H. Wang, and L. Shang. Thermal vs energy optimization for dvfs-enabled processors in embedded systems. In Proceedings of International Symposium on Quality Electronic Design, pages 204–209, 2007.

[58] A. Lungu, P. Bose, D. J. Sorin, S. German, and G. Janssen. Multicore power management: Ensuring robustness via early-stage formal verification. In Proceedings of IEEE/ACM International Conference on Formal Methods and Models for Co-Design, pages 78–87, 2009.

[59] J. P. Marques-Silva and K. A. Sakallah. GRASP: A Search Algorithm for Propositional Satisfiability. IEEE Transactions on Computers, 48:506–521, 1999.

[60] S. M. Martin, K. Flautner, T. Mudge, and D. Blaauw. Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads. In Proceedings of IEEE/ACM International Conference on Computer-Aided Design, pages 721–725, 2002.

[61] Marvell. Marvell StrongARM 1100 processor. Marvell Technology Group Ltd., 2004.

[62] A. Miller, A. Donaldson, and M. Calder. Symmetry in temporal logic model checking. ACM Computer Survey, 38(3):8, 2006.

[63] P. Mishra and M. Chen. Efficient techniques for directed test generation using incremental satisfiability. In Proceedings of International Conference on VLSI Design, pages 65–70, 2009. 163

[64] P. Mishra and N. Dutt. Graph-based functional test program generation for pipelined processors. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 182–187, 2004.

[65] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: engineering an efficient SAT solver. In Proceedings of Design Automation Conference, pages 530–535, 2001.

[66] C. Norstr ¨om, A. Wall, and W. Yi. Timed automata as task models for event-driven systems. In Proceedings of International Conference on Real-Time Computing Systems and Applications, page 182, 1999.

[67] M. Prasad, A. Biere, and A. Gupta. A survey of recent advances in SAT-based formal verification. International Journal on Software Tools for Technology Transfer (STTT), 7(2):156–173, 2005.

[68] X. Qin, M. Chen, and P. Mishra. Synchronized generation of directed tests using satisfiability solving. In Proceedings of International Conference on VLSI Design, pages 351–356, 2010.

[69] X. Qin and P. Mishra. Efficient directed test generation for validation of multicore architectures. In Proceedings of International Symposium on Quality Electronic Design, pages 276–283, 2011.

[70] X. Qin and P. Mishra. Automated generation of directed tests for transition coverage in cache coherence protocols. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 3–8, 2012.

[71] X. Qin, W. Wang, and P. Mishra. Tcec: Temperature- and energy-constrained scheduling in real-time multitasking systems. To appear in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2012.

[72] K. Sen and G. Agha. Cute and jcute: Concolic unit testing and explicit path model-checking tools. In Proceedings of International Conference on Computer Aided Verfication, pages 419–423, 2006.

[73] D. Shin and J. Kim. Dynamic voltage scaling of periodic and aperiodic tasks in priority-driven systems. In Proceedings of Asia and South Pacific Design Automation Conference, pages 653–658, 2004.

[74] S. Shukla and R. Gupta. A model checking approach to evaluating system level dynamic power management policies for embedded systems. In Proceedings of IEEE International High-Level Design Validation and Test Workshop, pages 53–57, 2001.

[75] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan. Temperature-aware microarchitecture: Modeling and implementation. ACM Transactions on Architecture and Code Optimization, 1(1):94–125, 2004. 164

[76] M. Song and S. Sahni. Approximation algorithms for multiconstrained quality-of-service routing. IEEE Transactions on Computers, 5(5):603 – 617, 2006.

[77] G. S. Spirakis. Designing for 65nm and beyond. In Keynote Address at the Conference on Design, Automation and Test in Europe, 2004.

[78] O. Strichman. Pruning techniques for the SAT-based bounded model checking problem. In Proceedings of IFIP WG 10.5 Advanced Research Working Conference on Correct Hardware Design and Verification Methods, pages 58–70, 2001.

[79] O. Strichman. Accelerating bounded model checking of safety properties. Formal Methods in System Design, 24(1):5–24, 2004.

[80] D. Tang, S. Malik, A. Gupta, and C. N. Ip. Symmetry reduction in SAT-based model checking. In Proceedings of International Conference on Computer Aided Verfication, pages 125–138, 2005.

[81] S. Vasudevan, D. Sheridan, D. Tcheng, S. Patel, W. Tuohy, and D. Johnson Goldmine: Automatic assertion generation using data mining and static analysis. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 626–629, 2010.

[82] R. Viswanath, V. Wakharkar, A. Watwe, and V. Lebonheur. Thermal performance challenges from silicon to systems. Intel Technology Journal, 4(3):1–16, 2000.

[83] I. Wagner and V. Bertacco. Mcjammer: adaptive verification for multi-core designs. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 670–675, 2008.

[84] S. Wang and R. Bettati. Reactive speed control in temperature-constrained real-time systems. In Proceedings of Euromicro Conference on Real-Time Systems, pages 10pp.–170, 2006.

[85] W. Wang and P. Mishra. Leakage-aware energy minimization using dynamic voltage scaling and cache reconfiguration in real-time systems. In Proceedings of International Conference on VLSI Design, pages 357–362, 2010.

[86] W. Wang and P. Mishra. Predvs: Preemptive dynamic voltage scaling for real-time systems using approximation scheme. In Proceedings of Design Automation Conference, pages 705–710, 2010.

[87] W. Wang, P. Mishra, and A. Gordon-Ross. Sacr: Scheduling-aware cache reconfiguration for real-time embedded systems. In Proceedings of International Conference on VLSI Design, pages 547–552, 2009.

[88] W. Wang, X. Qin, and P. Mishra. Temperature- and energy-constrained scheduling in multitasking systems: a model checking approach. In Proceedings of 165 .ACM/IEEE International Symposium on Low Power Electronics and Design, pages 85–90, 2010.

[89] Z. Wang and J. Crowcroft. Quality-of-service routing for supporting multimedia applications. IEEE Journal on Selected Areas in Communications, 14(7):1228 –1234, 1996.

[90] Z. Wang and S. Ranka. A simple thermal model for multi-core processors and its application to slack allocation. In Proceedings of IEEE International Symposium on Parallel and Distributed Processing, pages 1–11, 2010.

[91] J. Whittemore, J. Kim, and K. Sakallah. SATIRE: A new incremental satisfiability engine. In Proceedings of Design Automation Conference, pages 542–545, 2001.

[92] S. Williams. Icarus Verilog. Icarus Verilog, 2012. Available from: http:// iverilog.icarus.com/.

[93] D. Wood, G. Gibson, and R. Katz. Verifying a multiprocessor cache controller using random test generation. IEEE Design Test of Computers, 7(4):13 –25, 1990.

[94] F. Xie, M. Martonosi, and S. Malik. Compile-time dynamic voltage scaling settings: opportunities and limits. In Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation, pages 49–62, 2003.

[95] F. Xie, M. Martonosi, and S. Malik. Bounds on power savings using runtime dynamic voltage scaling: an exact algorithm and a linear-time heuristic approximation. In Proceedings of

International Symposium on Low Power Electronics and Design, pages 287–292, 2005.

[96] G. Xue, W. Zhang, J. Tang, and K. Thulasiraman. Polynomial time approximation algorithms for multi-constrained qos routing. IEEE/ACM Transactions on Networking, 16(3):656 –669, 2008.

[97] L. Yuan and G. Qu. Alt-dvs: Dynamic voltage scaling with awareness of leakage and temperature for real-time systems. In Proceedings of NASA/ESA Conference on Adaptive Hardware and Systems, pages 660–670, 2007.

[98] X. Yuan. Heuristic algorithms for multiconstrained quality-of-service routing. IEEE/ACM Transactions on Networking, 10(2):244–256, 2002.

[99] L. Zhang, C. F. Madigan, M. H. Moskewicz, and S. Malik. Efficient conflict driven learning in a boolean satisfiability solver. In Proceedings of IEEE/ACM International Conference on Computer-Aided Design, pages 279–285, 2001.

RAMALAKSHMI R did her bachelor of engineering in Electrical and Electronics Engineering at FRANCIS XAVIER ENGINEERING COLLEGE, Tirunelveli and doing Master of Engineering in Embedded System Technologies in SRI SHAKTHI INSTITUTE OF ENGINEERING AND TECHNOLOY, Coimbatore, Tamil Nadu, India. Her research interest includes Electronics and Embedded System Design.

RAJAMUNIPRIYA R did her bachelor of engineering in Electroincs and Communication Engineering at BANNARI AMMAN INSTITUTE OF TECHNOLOGY, Sathyamangalam and doing Master of Engineering in Embedded System Technologies in SRI SHAKTHI INSTITUTE OF ENGINEERING AND TECHNOLOY, Coimbatore, Tamil Nadu , India. Her research interest includes electronics and Embedded System Design.

[100] S. Zhang, K. Chatha, and G. Konjevod. Approximation algorithms for power minimization of earliest deadline first and rate monotonic schedules. In Proceedings of International Symposium on Low Power Electronics and Design, pages 225–230, 2007. 166

[101] S. Zhang and K. S. Chatha. Approximation algorithm for the temperature aware scheduling problem. In Proceedings of International Conference on Computer- Aided Design, pages 281–288, 2007.

[102] Y. Zhang, X. Hu, and D. Z. Chen. Task scheduling and voltage selection for energy minimization. In Proceedings of Design Automation Conference, pages 183–188, 2002.