Challenges for Future System-on-Chip Design

Thomas Hollstein* and Zebo Peng** and Raimund Ubar*** and Manfred Glesner*

Abstract – **Due to continuous improvements of semiconductor technologies new challenges for the design of highly integrated system-on-chip (SoC) solutions have arisen. Systems-on-Chip provide an implementation platform for many applications, and will revolutionize the design of future electronic systems. In this contribution we focus on several important challenges and unsolved problems concerning SoC design and testability.**

1 Introduction

The speed of the evolution of semiconductor processes and the demand for complex and highly integrated applications have led to the need for very efficient future design methodologies in order to handle the complexity of the design and to satisfy time-to-market constraints in the future. For the year 2015 the SIA roadmap predicts a density of 2 billions of transistors per chip for ASIC technologies and for DRAMs a density of 48 GBits per Chip[1]. Currently an integration density of 30-40 million transistors per chip is available and this already enables an system-level on-chip integration. In the field of digital design, system-on-chip (SoC) solutions already become state-of-the-art. Currently SoCs are mainly realized by integrating several pre-designed cores on one and the same die. Typically components like processor cores, dedicated ASIC blocks, interfaces and memories are combined to form hardware/software solutions. Nevertheless the design of SoCs raises a lot of EDA problems, including adaption of blocks (inter-block communication structures and protocols), testability, realization of low-power, and performance requirements with respect to the whole system to be integrated.

The trend of system-level integration will continue in the future and this will have a strong influence on the designer's environment. Increasing technology capabilities will enable new qualities of system integration: the analog and digital world will grow together and even micro-mechanic systems are potential SoC components (e.g. sensors). The demand for advanced mobile applications and distributed intelligent sensor arrays will lead to an integration of RF components (to be realized in CMOS) into SoCs. The basic challenges for SoC design arise from the heterogeneity with respect to the combined components. This has an influence on the design and validation. Another serious problem is the post-fabrication test of SoC devices and, derived from that, the question how systems can be designed in order to be testable.

In the following, we focus on design challenges for digital SoCs: First advanced system design methods which are able to cope with the system's complexity (IP-based design (IP="Intellectual Property", decribed in next section) and system synthesis techniques) are discussed. Then the testability issue will be addressed: SoC testing and efficient generation of testing data. Finally we'll draw some conclusions combined with an outlook to future trends.

2 System Design: IP-based Design

The reuse of components, designed for a class of applications, is a method to reduce the design-effort, which is well-known from software design for a long time already. In the field of ASIC design, the reuse of blocks has been practiced in design houses mainly in form of an evolution of existing products. Due to shorter product cycles and rapidly increasing product complexity, many design companies will more and more refer to module cores from outside. During the process of the transfer of design blocks from the original provider to the integrator, intellectual property (IP) issues have to be considered. In [2] some essential issues for IP reuse are outlined: design quality, documentation, security, support, and integration.

Figure 1 outlines an IP-based design flow. The specification of new IPs is performed based on demands arising with new product classes to be realized. During the specification the spectrum of applications for the IP integration has to be evaluated and the required generality of the IP block to be developed has to be determined. Based on the IP specification, an IP model is developed (Soft IP). A precise and well-structured documentation is essential for future modifications and enhancements of the block. The IP-model can directly be distributed as soft IP or it will be synthesized by the IP provider for a dedicated target technology (resulting in an hard IP). The distribution of the IP can be performed directly by

* Darmstadt University of Technology, Institute of Microelectronic Systems, Karlstr. 15, 64283 Darmstadt, Germany. E-mail: thomas@mes.tu-darmstadt.de, Tel: +49-6151-16-4038, Fax: +49-6151-16-4936., ** Linköping University, Dept of Computer Science, SE-581 83 Linköping, Sweden. E-mail: zpe@ida.liu,se, Tel: +46-13-282067, Fax: +46-13-282666, *** Tallinn Technical University, Raja 15, 12618, Tallinn, Estonia. E-mail:raiub@pld.ttu.ee, Tel: +372 620 2252, Fax: +372 620 2253.

the provider or by an IP library provider. For the integration of an IP block into the customer's product design, support services have to be provided.

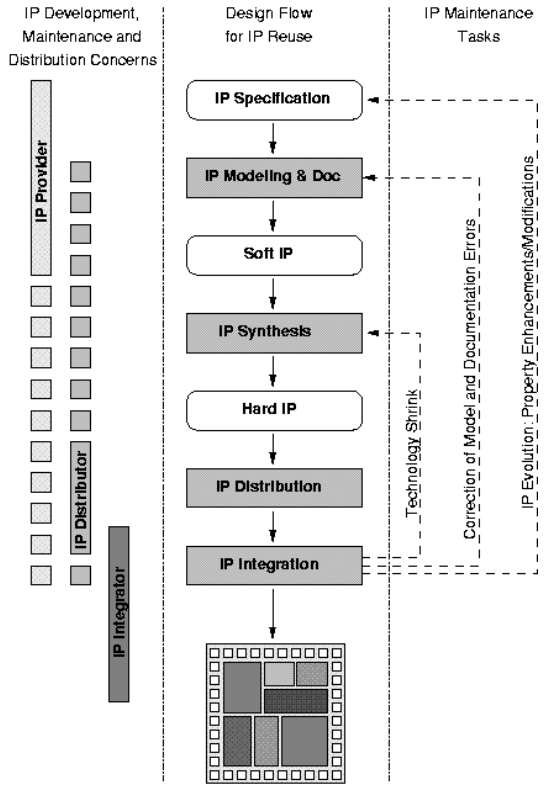

Figure 1: IP-based design flow

In this flow, a tight cooperation of IP provider, distributor and customer is required in order to exploit the potential improvement of efficiency. Concerning the efficiency gain of IP-based design some drawbacks of the method have to be considered: re-usability of design components generates a demand for generality. Generality often comes along with a loss of performance and an increased space to be covered in circuit test. Furthermore, permanent technology improvements necessitate regular re-synthesis of the IP blocks to modified technologies. Typically this cannot be done automatically, since constraint/performance trade-offs require an adaptation of the synthesis parameters. Supplementary IP model modifications can be very time-consuming and can typically not be performed without the knowledge of the original designers. The maintenance effort for IP modules can be comparatively high and therefore the reuse concept is well-suited for large IP blocks (coarse granularity) only. This is for example the case for the ARM microprocessor cores [3]. Another drawback of IP based design is the synchronization and the communication in between of IP blocks. The latter normally have been designed and optimized for a specific clock period. The synchronization of different clocks and the clock distribution on large dies is a difficult task as well as the design of appropriate communication structures.

Increasing improvements of CMOS technologies also enable the integration of analog components into mixed-signal SoCs. Since synthesis techniques for analog ICs are still outstanding, the reuse concept is of basic interest for the integration of analog and radio frequency components in embedded single-chip applications.

The SoC integration of IP blocks contains a lot of challenges for future research: communication networks, testability, power minimization. If the IP blocks to be integrated operate with different clock frequencies, in most cases asynchronous communication protocols have to be implemented, which will lead to a testability nightmare. The testability aspects will be discussed in section 4 in detail. Furthermore, the design goals in terms of performance and power minimization have to be realized not only for single blocks, but also for any combination of blocks. Out of these currently more or less unsolved problems a lot of research topics will arise in the future.

3 System Synthesis Techniques

Powerful system synthesis techniques can deliver a second strategy to cope with future system's increasing complexities. In the recent years, several methods have been evolved from high-level synthesis to system-level synthesis. Research in this context has been focused on system-level specification methods and hardware/software co-design, where several partitionings between hardware and software parts of the specification are exploited. The main challenges of system synthesis techniques are the simulation of heterogeneous systems, the handling of the design space complexity and estimation issues as well as the inclusion of the designer's know-how into the design process [4][5].

Furthermore the migration of design representations is still a major issue. In many domains, behavioral system models are written in C(++), using floating point number representations. The hardware implementation requires fix-point numbers in order to obtain cost efficient implementations. Another problem is the selection of application algorithms. An algorithm designed for software implementations may be sub-optimal for hardware realizations and vice versa. In particular, the migration of software specifications using pointer operations to hardware can be very complicated. New unified hardware/software specification approaches like SystemC [6] can be a platform for solving these problems, since they provide a unified specification semantics.

4 Testing of Systems-on-Chip

While the core-based design technique, discussed in section 2, has led to increased design productivity, it introduces additional test-related problems, which are due to, among others, intellectual property protection. These additional testing problems, together with the test problems induced by the complexity and heterogeneous nature of SoC, pose great challenges to the SoC testing community [7]. This section will discuss several of these challenges.

It should first be noted that, even though the core-based design strategy is similar to traditional system-on-board (SoB) design where individual chips are designed and then integrated into a board, production test of SoC and that of SoB are very different. In SoB testing, the individual chips are manufactured and tested first before they are integrated into the board. The individual SoC cores, while pre-design and pre-verified, will not be tested until they are integrated into a system chip. Therefore a core is not tested individually, but rather as a part of the overall system chip test. This means that the divide-and-conquer testing strategy traditionally used to deal with the complexity of testing a complex board cannot be directly applied in SoC testing.

Besides the increased complexity, the difficulty of SoC testing is due to its heterogeneous nature. Typically, a SoC consists of microprocessor cores, digital logic blocks, analog devices, and memory structures. These different types of components were traditionally tested, as separate chips, by dedicated automatic test equipment of different types. Now they must be tested all together as a single chip either by a super tester, which is capable of handling the different types of cores and is very expensive, or by multiple testers, which is very time consuming due to the handling time of moving from one tester to another.

Another problem related to testing embedded cores as a part of system test is due to the limited knowledge the system integrator has about the internal structure of a core. This may be due to intellectual property protection or the use of complex hard cores. In this situation, the core developer will provide the test patterns and insert design-for-test (DFT) mechanism into the core. Since the core developer has no idea about the overall SoC design and test strategy to be used, the inserted DFT mechanism may not be compatible with the overall design and test philosophy, leading usually to low test quality or high overhead. This problem needs to be solved in order to guarantee the high quality level of SoC products.

Another key issue to be addressed for SoC testing is the implementation of test access mechanism on chip. For traditional SoB design, direct test access to the peripheries of the basic components, in the form of separate chips, is usually available. For the corresponding cores embedded deeply in a SoC, such access is impossible. Therefore, additional test access mechanism must be included in a SoC to connect the core peripheries to the test sources and sinks, which are the SoC pins when testing by external tester is assumed.

The design of the test access mechanism must be considered together with the test-scheduling problem, in order to reduce the silicon area used for test access and to minimize the total test application time, which includes the time to test the individual cores and user-defined logic as well as the time to test their interconnections. The issue of power dissipation in test mode should also be considered in order to prevent the chip being damaged by over heating during test. Since the problems of test access mechanism design, test scheduling, test application time minimization, and test power consideration are interdependent on each other, they must be solved together in an integrated design environment [8].

Many of the testing problems discussed above can be overcome by using build-in self-test (BIST) strategy. For example, the test access cost can be substantially reduced by putting the test sources and sinks next to the cores to be tested. BIST can also be used to deal with the discrepancy between the speed of the SoC, which is increasing rapidly, and that of the tester, which will soon be too slow to match typical SoC clock frequencies. The introduction of the BIST mechanism in a SoC will also improve its diagnosisability and field-test capability, which are essential for many applications where regular operation and maintenance test is needed.

Since the introduction of BIST mechanism into a SoC is a complex task, we need to develop powerful automated design methods and tools to optimize the test function together with the other design criteria as well as to speed up the design process.

As discussed earlier (section 3), SoC design means often hardware/software codesign,. The testing of the hardware and software parts of a HW/SW system are still considered as separate problems and solved with very different methods. There is great need for a general strategy to deal with hardware and software testing in a systematic manner so that the testing cost can, for example, be considered in the hardware/software partitioning process

5 Hierarchical Test Generation

Automated test pattern generation for complex digital systems encompasses three main activities: selecting a method for modeling the system, developing a fault model and generating tests to detect all the faults covered by the fault model. The efficiency of test generation (quality of tests, and speed of test generation) is highly depending on the level of system representation and fault models which have been chosen.

Due to the increasing complexity of digital circuits the classical gate-level methods have become impractical. Hence, other approaches based mainly on functional,

behavioral, or hierarchical methods are gaining more and more popularity [9][10]. However, functional and behavioral test synthesis methods, which do not use implementation data cannot afford good test quality measured in terms of low-level faults or defects. As a possible solution, hierarchical methods have evolved [10] which take advantage of higher abstraction level information while generating tests for the gate-level faults. The advantage of hierarchical approaches compared to the functional ones lies in the possibility of constructing test plans at higher functional levels, and modeling faults at lower levels.

The traditionally used very popular stuck-at fault (SAF) model has not withstood the test of time. It has been shown that high SAF coverage cannot quarantee high quality of testing, for example, for CMOS integrated circuits. The reason is that the SAF model ignores the actual behaviour of circuits, and does not adequately represent the majority of real IC defects and failure mechanisms which often do not manifest themselves as stuck-at faults. This facts is well known but usually ignored in the engineering practice because of the complexity issue.

Good results in gate level test generation have been achieved with Decision Diagrams (DD) as the model of digital circuits. Recent research has shown that DDs can be very efficiently used also at higher levels providing a uniform model for both gate and register transfer level test generation [11]. Using DDs affords easily to adopt classical gate-level fault activating, fault propagation and line justification algorithms in higher level test generation.

In [12] a new approach was introduced for hierarchical defect-oriented test generation based on defect preanalysis for components, and using the results of preanalysis in higher level fault simulation and test generation. By introducing a functional fault model this idea is generalized as a method for mapping faults from one hierarchical level to another. The functional fault model in a form of a set of logical conditions allows to logically represent the defects in components and in the communication networks by a uniform technique.

Combining the efficiency of high register-transfer level test planning and the accuracy of the medium gate-level fault "transportation" analysis with low-level exact physical defect activation allows to reach high efficiency in test generation with high test quality on the other hand.

6 Conclusions

Due to rapidly improving ASIC technologies and a permanent application-driven demand for more and more complex system integration, advanced system design methodologies are mandatory in order to achieve a sufficient design efficiency. In this contribution we have described future challenges for IP-based and synthesis-driven SoC design. New methods for test generation and design for testability for complex systems-on-chip will also be a major issue in future SoC research.

References

[1] SIA, "Int. Technology Roadmap for Semicond. 1999/Update 2000", http://public.itrs.net/.

[2] Gajski, D.D.; Wu, A.C.-H.; Chaiyakul, V.; Mori, S.; Nukiyama, T.; Bricaud, P., "Essential Issues for IP Reuse", Proceedings of the ASP-DAC 2000, Page(s): 37 –42

[3] http://www.arm.com

[4] S. Vercauteren, B. Lin, H. de Man, "Constructing Application Specific Heterogeneous Embedded Architectures from Custom HW/SW Applications, DAC '96, pp. 521-526

[5] T.Hollstein, J. Becker, A. Kirschbaum, M. Glesner, "HiPART: A New Hierarchical Semi-Interactive HW/SW Partitioning Approach with fast Debugging for Real-Time Embedded Systems, CODES/CASHE '98

[6] http://www.systemc.org

[7] Y. Zorian, E. J. Marinissen and S. Dey, "Testing Embedded-Core Based System Chip, *Proc. International Test Conference*, 1998.

[8] E. Larsson and Z. Peng, An Integrated System-On-Chip Test Framework, *Proceedings of Design, Automation and Test in Europe*, 2001.

[9] J.Lee and J.H.Patel. ARTEST: An Architectural Level Test Generator for Data Path Faults and Control Faults. *Proc. Int. Test Conf.* Oct. 1991, pp. 729-738.

[10] E.M.Rudnick, R.Vietti, A.Ellis, F.Corno, P.Prinetto, M.Sonza Reorda, "Fast sequential circuit test generation using high-level and gate-level techniques", *Proc. of DATE*, 1998.

[11] J.Raik, R.Ubar. Fast Test Pattern Generation for Sequential Circuits Using Decision Diagram Representations. Journal of Electronic Testing: Theory and Applications. Kluwer Academic Publishers. Vol. 16, No. 3, pp. 213-226, 2000.

[12] R.Ubar, W.Kuzmicz, W.Pleskacz, J.Raik. Defect-Oriented Fault Simulation and Test Generation in Digital Circuits. 2nd Int. Symp. on Quality of Electronic Design, San Jose, California, March 26-28, 2001..