International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-3, Issue-5)

Research Article | **May 2014**

Advance Microcontroller Bus Architecture Based System on Chip Platform to Minimize Energy Consumption in Portable Devices

Mr.A.Sajeevikumar[1], Dr.G.Gunesekaran[2] , Mr.V.Prabhu[3] ,
[1]Research Scholar MAHER University, [2]Professor and Principal Meenakshi College of Engineering
[3]Research Scholar MAHER University, India

Abstract—

Over the past ten years, as integrated circuits became increasingly more complex and expensive, the industry began to embrace new design and reuse methodologies that are collectively referred to as system-on-chip (SOC) design. In this paper, we focus on the reuse and integration issues encountered in this paradigm shift. The reusable components, called intellectual property (IP) blocks or cores, are typically synthesizable register-transfer level (RTL) designs (often called soft cores) or layout level designs (often called hard cores). The concept of reuse can be carried out at the block, platform, or chip levels, and involves making the IP sufficiently general, configurable, or programmable, for use in a wide range of applications. The IP integration issues include connecting the computational units to the communication medium, which is moving from ad hoc bus-based approaches toward structured network-on-chip (NOC) architectures. Design-for-test methodologies are also described, along with verification issues that must be addressed when integrating reusable components.

Keywords— ***Analog intellectual property (IP); intellectual property (IP) cores; network-on-chip (NOC); platform-based design; programmable intellectual property (IP); system-on-chip verification; system-on-chip (SOC).***

I. INTRODUCTION

Today's System-On-Chip (SOC) devices are targeting complex multi-media applications where there is a need for significant amount of Digital Signal Processor (DSP) computing power in order to achieve the various data processing tasks which could include both video and audio. In order to complete these tasks various DSP Intellectual Property (IP) cores, which are optimised for the application in hand, are utilised. In order to satisfy the constraint of fast time to market effective platform architectures are required. High performance IP cores can be connected to the above platforms on a plug-in basis in order to maintain performance constraints as well as that of the fast time to market [1][2].
Although numerous researchers in the literature have considered the development of interfaces and wrappers [3][4], only a few discuss implementation issues and impact at the SOC level. This paper will discuss the implementation of an efficient interface strategy and its effect on the overall system. The paper describes the development of an SOC platform for low power wireless applications. The platform utilises the LEON Processor, which is a general SPARC compatible processor [5].
In order to integrate a number of DSP IP cores, an interface is required. This interface connects the DSP IPs directly to the Advance Microcontroller Bus Architecture (AMBA)[6]of the LEON Processor. In comparison to the bus interface design that contains a Virtual Component Interface (VCI) , the AMBA interface described in this paper eliminates the VCI overhead. As discussed in[7]. it is less efficient to design extra VCI for both the bus interface and DSP IPs. Therefore, the AMBA interface discussed in this paper connects the DSP IPs directly to the processor without any additional control signals. This is achieved through gated clock and look ahead features. At the same time, these features have made the interface consumed less power.
Gated clock has been employed normally to selectively stop the clock in portions of circuits where active computation is not being performed [9][10]. The Finite State Machine (FSM) is partitioned into several sub-FSMs and these sub-FSMs are clocked when necessary. However, the AMBA interface discussed in this paper is partitioned into Bus Address and Control Decoders, an AHB Data FSM and a Clock Controller. The AHB Data FSM is clocked when there is data available from the processor and the selected DSP IP is clocked when the data is available from the interface.
The DSP IP core used in this paper is an FIR filter that has a transpose direct form structure [11][12]. This structure reduces the switching activity at data inputs of the multiplier. The data input of the multiplier remains unchanged until it is multiplied by all the coefficients. In addition, the transpose direct form structure can be exploited by different numerical ordering algorithms for manipulating filter coefficients in order to reduce the switching activity at the coefficient input of the multiplier.

II. REUSEABLE IP

The main prerequisite for creating SOC designs is a set of reusable IP blocks that support plug-and-play integration. IP blocks are the highest level building blocks of an SOC. A library of reusable IP blocks with various timing, area, power configurations is the key to SOC success as it allows mix-and-match of different IP blocks so that the SOC integrator can apply the tradeoffs that best suit the needs of the target application. The process of creating a reusable IP block, however, differs from the traditional ASIC design approach. Typically, it may require five times as much work to generate a reusable IP block compared to a single-use block[14]. Details on the IP design process may be found in a wide variety of

sources. However, the most noteworthy reference on the subject is the Reuse Methodology Manual (RMM)[14]. which provides a comprehensive set of guidelines on the reusable IP design process. In the sections to follow, we provide an overview of the issues in design issues for reusable digital IP, analog IP, and programmable IP.

A. Digital IP

Digital IP blocks are the most popular and ubiquitous form of reusable IP in industry today. Many of the tools and design methodologies for creating digital IP were already in place when the concepts of reusability emerged. However, there have been wholesale changes in the design flow to fully enable reusable design. In addition, there are many technical issues that need to be addressed, as the IP developer must anticipate all of the applications in which the IP block may be used. The technical issues associated with the design will not be repeated here, since they are well documented in the RMM[14].

The three stages in the design process are as follows:

• Specification and documentation of the reusable IP;

• Implementation using standardized coding practices;

• Full verification including code coverage and behavioral (or functional) coverage.

The first step includes the generation of suitable documentation for the IP block. The second step includes code design, synthesis, and design for test. Somewhat surprisingly, the second step of implementation is only a small part of the reusable IP design process. In fact, depending on the size and type of the IP, the third step of verification may take up to 50% of the total time. Since the IP may be reused many times in a variety of unanticipated applications, design errors within the block cannot be tolerated. The goal of verification is to achieve 100% code coverage and close to 100% functional coverage to ensure the IP block works correctly.

The actual form of a reusable IP core can vary de-pending on the way in which the IP developer/vendor chooses to provide the core to the system designer. There are three main categories: [14] soft, firm, and hard. These forms are described below and their relationships and tradeoffs are depicted in Fig. 1.

• Soft IP blocks are specified using RTL or higher level descriptions. They are more suitable for digital cores, since a hardware description language (HDL) is process-independent and can be synthesized to the gate level. This form has the advantage of flexibility, portability, and reusability, with the drawback of not having guaranteed timing or power characteristics, since implementation in different processes and applications produces variations in performance. Sometimes, the HDL is obtained from a third party in an encrypted form which does not allow it to be modified. This makes it harder to adapt it for use in an unanticipated way, but it also prevents the user from introducing any new design errors into the block.

• Hard IP blocks have fixed layouts and are highly optimized for a given application in a specific process. They have the advantage of having predictable performance. This, however, comes with additional effort and cost, and lack of portability that may greatly limit the areas of application. This form of IP is usually prequalified, meaning the provider has tested it in silicon. This adds some assurance to its correctness.

• Firm IP blocks are provided as parameterized circuit descriptions so that designers can optimize cores for their specific design needs. The flexible parameters allow the designers to make the performance more predictable. Firm IP offers a com-promise between soft and hard, being more flexible and portable than hard IP, yet more predictable than soft IP.

Until very recently, most digital IP blocks came in the form of hard IP. For example, ARM and MIPS core vendors would provide behavioral models and black-box layouts of the processors for use during design and verification, and then B drop in the hard IP at the foundry facility before fabrication. This afforded the vendor some level of IP protection while allowing the customer to carry out designs using the IP. More recently, soft IP has become the preferred handoff level. Typical soft IP (cf. [109]) include interface blocks (USB, UART, PCI), encryption blocks (DES, AES), multimedia blocks (JPEG, MPEG 2/4), net-working blocks (ATM, Ethernet), and microcontrollers (HC11).

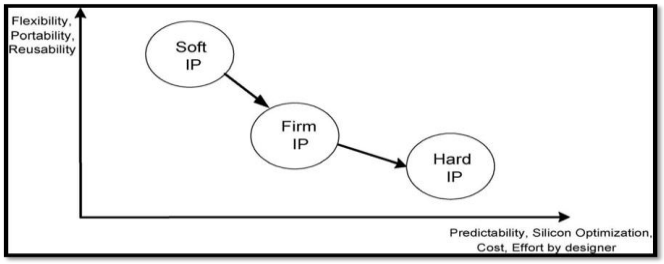

Figure 1.Different types of IP blocks

The RTL descriptions usually are configurable in that certain parameters are user-definable to tailor the block to the needs of the customer. This allows selective inclusion or exclusion of distinguishing features which can impact the final implementation's performance, cost, and power. In the case of a processor core, parameters such as the bus width, number of registers, cache sizes, and instruction set may vary from customer to customer, so the flexibility of soft IP allows for their modification before synthesis. Commercial tools exist for this purpose in which a configurable processor with specific attributes can be automatically generated[25]. Such flexibility provides lower area, power, and improved performance, since the IP block can be tuned for each application. In addition, some features in a given IP block that are not needed by all customers can be removed. With some effort, the final design generated from a soft IP block can be within 20%– 30% of the power and timing of a hard IP implementation.

B. Analog/Mixed-Signal Design for Reuse

While design productivity can be improved significantly with the use of digital IP blocks, another bottleneck exists if the designs include analog and mixed-signal components. Digital design has a well-defined, top-down design methodology but analog/mixed-signal (AMS) design has tradition-ally been an ad hoc custom design process. When analog and digital blocks coexist on the same substrate, the analog portion of the design can be more time-consuming to develop even though it may represent a smaller percentage of the chip area. In a few years, it is anticipated that more than 70% of all SOC designs will have some form of analog/mixed-signal blocks [13]. This increase is consistent with the expected growth of the wireless industry over the same period.

In order to keep pace with rapidly evolving markets, the productivity of AMS design can be improved using a mixed-signal SOC design flow,[20] employing AMS IP. One of the main advantages of the use of AMS IP in SOCs is the potential reduction in power, which is especially important in battery-operated applications such as personal digital assistants (PDAs), wireless local area networks (LANs), etc. Typical AMS components include operational amplifiers, analog-to-digital converters (ADCs), digital-to-analog converters (DACs), phase-locked loops (PLLs), delay-locked loops (DLLs),serializer/deserializer transceivers, filters, voltage references, radio-frequency (RF) modules, voltage regulators, analog comparators, etc. Many of these blocks are delivered in the form of hard IP and targeted to one application in a specific fabrication technology. Currently, because of the complexity of analog/mixed-signal design and its sensitivity to the surrounding environment, AMS blocks are most commonly presented in the form of hard IP. However, this form has limited scope of applications. Hard IP will reduce the design cycle significantly when the specifications and fabrication processes are identical, but will not greatly improve the design cycle if it has to be modified in any way or migrated to a new process. This calls for a more flexible definition for the format in which the AMS IPs are provided. Firm IP appears to be the most appropriate format to deliver the AMS IP library components. In this form, the IP captures suitable schematics of the analog blocks with parameters that are adjustable to optimize the design for specific applications. Unlike hard IP, this form allows ease of migration of IP from foundry to foundry, customer to customer, and application to application.

Since AMS blocks cannot be easily synthesized from a high-level specification without low-level support, designers must follow a design process such as the firm IP hardening flow [15] illustrated in Fig. 4. The starting point of the flow is the set of selected library components that comprise the un optimized schematic view of the design. This library consists of parameterized reusable components and is an essential part of the design flow. The model parameters are set by an optimization tool to achieve the derivative design specifications. After an architecture is chosen, the firm IP is taken through the IP hardening flow for optimization of the circuit parameters to maximize
performance and to generate the final GDSII layout of the block. Proper modeling of the interfaces between the different blocks is important in the design process to account for different effects such as loading and coupling. This is needed to achieve correct performance when used in the overall system-level design[21].

It is clear that the development of AMS IP must take a different approach compared to digital IP development. The IPs must be able to handle and transfer both design experience and heuristics from the original design to subsequent design derivatives. In reality, this constitutes the reusable IP in the analog design process.

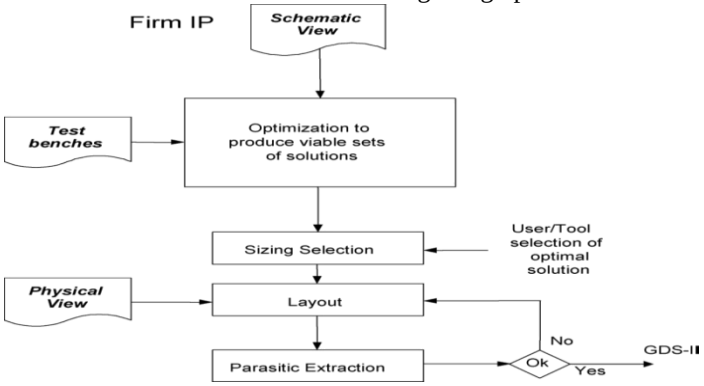

Figure 2. PROPOSED ANALOG/MIXED-SIGNAL IP HARDENING FLOW.

C. Programmable IP

As s become larger and more complex, the design costs are so high that it becomes important to incorporate programmability within the SOC to allow for reuse at the chip level. This programmability can appear in a number of forms: hardware programmability using programmable logic cores and software programmability using an embedded processor. The key to programmable SOC designs is to provide some form of flexible hardware and or software infrastructure, often called the programmable fabric. We distinguish between two types of flexibility: prefabrication and post fabrication.

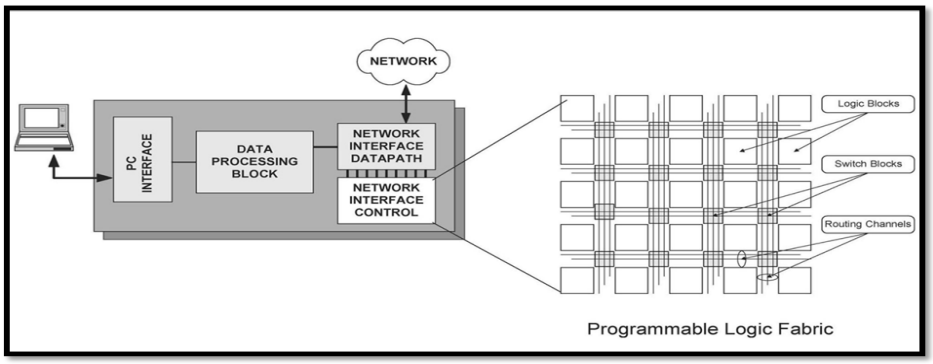

Figure 3. SOC APPLICATION CONTAINING A PROGRAMMABLE LOGIC FABRIC.

Prefabrication flexibility involves the use of a structured ASIC fabric [22][23]. In this approach, the lower layout layers of the structured ASIC fabric are predetermined and chips are already partially fabricated. The programming is performed by modifying the upper metal and via layout layers just before final fabrication. Since only a few steps are performed to complete fabrication, the turnaround time is relatively short. With this approach, however, the design cannot be changed after the chip is manufactured.

Post fabrication flexibility allows the behavior of an IP block to be modified after the chip has been fabricated, by either rewriting software/firmware to run on an embedded processor (software configurability) or by reconfiguring embedded programmable hardware, which involves changing the state of configuration bits embedded in the fabric (hardware programmability).

1) Hardware Programmability: Hardware programmability is enabled by the incorporation of one or more programmable logic cores into an SOC. The programmable logic core is a flexible logic fabric that can be customized to implement any digital circuit after fabrication. Such an embedded core may look very much like the programmable fabric in a stand-alone field-programmable gate array (FPGA)[24]. Before fabrication, the de-signer embeds the fabric (consisting of many uncommitted gates and programmable interconnects between the gates) onto the chip. After the fabrication, the designer can then program these gates and the connections between them. Fig.2 shows a hypothetical SOC containing such a core. In this example, the PC interface, the data processing block, and the network interface data path are implemented using fixed-function IP blocks. However, rather than implementing the network interface control functions using fixed logic, the SOC designer embeds a generic programmable logic core onto the IC. The fabric consists of logic blocks and routing channels with switch blocks at their intersections. After fabrication, the network interface control functions are mapped to and implemented on the programmable logic fabric. This post fabrication mapping is typically done using automated tools that are different than the tools used to implement the SOC before fabrication. The internal structure (architecture) of programmable logic cores often inherits much from the architecture of stand-alone FPGAs. as described earlier, IP cores can either be hard cores or soft cores, depending on whether the core is delivered as a layout or in the form of a hardware design language (HDL) description. The same is true for programmable logic cores. The programmable logic cores described above are hard cores. A soft programmable logic core is described in; in this architecture, the behavior of the core is described using an HDL (note that this is different than the behavior of the circuit to be implemented in the core). The soft core makes integration simpler, since standard SOC tools can be used. However, the overhead for a soft programmable logic core is typically six times larger than that for a hard programmable logic core, meaning it is only suitable if only small amounts of programmable logic are required. Development of automated hard-core generators for programmable logic cores is also a continuing area of research.

2) Software Programmability: Software is the most natural form of programmability for SOC. Embedded software, allows a single SOC to have different B personalities and serve different customers or market segments. From a design perspective, as much of the solution should be implemented in software as possible to maximize the flexibility of the design. Complex control functions are better suited to software implementations whereas data operations are better implemented in hardware. If performance is an issue, then hardware must be used for the implementation which can take

the form of a programmable logic fabric, which is roughly five times faster than software, or an ASIC implementation, which is typically 50 times faster than software (based on the authors' experience).
Reuse in software is provided through the use of libraries of code and data structures, along with off-the-shelf kernel and real-time operating systems (RTOSs) to improve productivity. Unlike hardware, the software or firmware is never actually completed; it is under continuous development but the code is frozen and released as a version with the intention to update it in the future (and perhaps provide bug fixes when needed). Since SOC programmability is headed in this direction, there will be a growing need for more software/firmware designers in an industry typically dominated by hardware engineers.

III . PLATFORM-BASED DESIGN

Productivity gains obtained strictly from reusable IP has its limits, since the process of chip integration can be very time-consuming and expensive even with predesigned blocks. What is needed to improve productivity is a higher level of design abstraction. Economies of scale can be derived from the fact that a single architecture may serve many different customers in one market segment. Furthermore, as the needs of a marketplace change, it should be possible to use a configurable architecture as a basis for new designs. For these reasons, the concept of platform-based design[16] was introduced where new designs could be quickly created from an original platform to amortize costs over many design derivatives. More specifically, a platform is an abstraction level that covers a number of refinements to a lower level. It allows significant improvements in designer productivity, since many high-level and low-level decisions have already been incorporated into the platform, and all associated tools and flows are in place to quickly generate new designs. The processor requires high-speed connections to certain blocks such as the shared memory controller (SMC), the direct memory access (DMA) controller, the test interface controller (TIC), and the power and clock control units. The connections are implemented using the synchronous advanced high-speed bus (AHB) of the AMBA bus standard [17]. This bus requires the development of a bus arbiter and decoder functions. The other components in the system operate at different speeds and are connected using the asynchronous advanced peripheral bus (APB). These components include the watchdog timers, other timers, general-purpose input/output (GPIO) devices, and programmable interrupt controller (PIC), UART, and USB interface standards. The baseband controller (BBC) that performs the DSP functions on the incoming and outgoing bit streams and radio control functions is also interfaced to the APB. A bus bridge is used to interface the AHB to the APB. The data received from the ADC and transmitted out through the DAC (and eventually to/from the RF module) would be implemented as a separate chip in this case.

IV. ON-CHIP COMMUNICATION INFRASTRUCTURE

A typical SOC today consists of many cores operating at different clock frequencies and this presents additional issues for IP integration. For example, digital signal processors (DSPs) and network interface cores include circuits whose clock frequencies must match the sample rate or bit-rate of the data that they are processing. Apart from these types of constraints, different cores may be designed to operate at different clock rates, a situation that is nearly unavoidable when cores are obtained from several sources, both internal and external to a company.
Design flows based on traditional logic synthesis assume a synchronous implementation: all timing issues are regulated by a single, global clock. More recently introduced system level design languages such as System-C [25] and System Verilog[26] allow designs to be described as communicating processes by using transactional models. With this decoupling, each core in an SOC design can be viewed as a separate synchronous island, and the interfaces between these islands are provided by the chip's global interconnect. This approach is commonly referred to as globally asynchronous, locally synchronous (GALS) design. The work in adapted the GALS idea to define B latency insensitive designs, wherein the inter-connect can be either synchronous or asynchronous. The key idea in this approach is to restrict the system-level design to one where correct functionality can be guaranteed regardless of the latency of links between cores. This allows the interconnect to be pipelined and provides a separation of timing and functionality similar to that of traditional synchronous design. However, latency remains critical for the performance of many SOC designs. Integrating latency and performance requirements into an SOC design flow remains an area of active research. Synthesizing designs from multi-timed, transactional descriptions is another topic of ongoing research A. Network-on-Chip

A more structured interconnect fabric is being pursued in the research community for commercial de-signs that must integrate a large number (10–100) of IP blocks in a single SOC. Today, there exist many SOC de-signs that contain a number of processors in applications such as set-top boxes, wireless base stations, HDTV, mobile handsets, and image processing[27]. Such systems behave as multiprocessors, and require a corresponding design methodology for both their hardware and software implementations. Power budgets and cross-chip signaling constraints are forcing the development of new design methodologies to incorporate explicit pipelining and pro-vide a more structured communication fabric. Many researchers have suggested that future designs will be dominated by arrays of processors that form the basis of new multiprocessor SOC platforms (the so-called MP-SOC platforms)[27]. Network-on-chip (NOC), infrastructures are emerging as the new paradigm that characterizes the on-chip data communication architecture of large-scale SOCs. The integration of several components into a single system gives rise to new challenges. As shown in Fig. 9, there are a wide variety of topologies that have been proposed for NOC including SPIN, CLICHE, torus, folded torus and irregular and bufferfly fat-tree. The practical implementation and adoption of the NOC design paradigm faces various unsolved issues

related to design methodologies, test strategies, and system reliability.

B. Design Considerations

It is well known that it can take several clock cycles for a global signal to travel from one end of a chip to the other. To cope with this issue, the end-to-end communication medium can be divided into multiple pipelined stages, with delays in each stage comparable with the clock-cycle budget. In NOC architectures, the inter-switch wire segments together with the switch blocks constitute a highly pipelined communication medium characterized by link pipelining, deeply pipelined switches, and latency-insensitive component design. Link pipelining is inherently built-in to regular NOC topologies. The switch designs generally consist of multiple pipeline stages and the delay of each stage is less than the target clock period in a particular technology node.

V. SOC TEST METHODOLOGIES

Another important aspect of SOC integration is the development of a test methodology for post manufacturing tests. Core testing strategies often accompany a third-party IP block when it is purchased or otherwise acquired. However, system-level test integration is left to the SOC platform designer.

A. IP Core Level Test

The test of an IP core typically consists of internal DFT structures and the required set of test patterns to be applied and captured on the core periphery. The test pat-terns need to include data and protocol patterns. The data patterns contain the actual stimulus and response values, whereas the protocol pattern specifies how to apply and capture the test data. The core internal test should be carried out by the core provider, as the system integrator, in most cases, has very limited knowledge about the structural content of the adopted core and hence considers it as a black box. It may not be possible for the system integrator to prepare the necessary test for it. Consequently, the core creator should be responsible for delivering: 1) the DFT hardware inside the core; 2) the test patterns of the core; and 3) the validation of those test patterns.

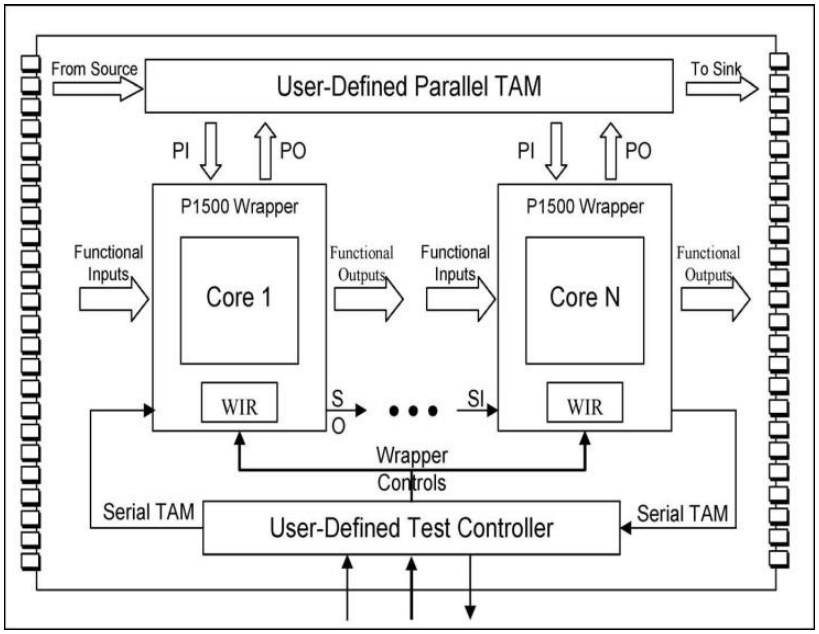

Figure 4. . CORE-BASED SOC TEST ARCHITECTURE

B. SOC Level Test

A conceptual architecture suitable for testing embedded core based SOCs is presented in Fig. 11, and has three principal components as explained below.

1) Test Pattern Source and Sink: The test pattern source generates the test stimuli and the sink receives the test responses.

2) Test Access Mechanism (TAM): The test access mechanism performs the on-chip test pattern transport. It can be used to transport either test stimuli from the test pattern source to the core under test or to transport the test responses from core under test to a test pattern sink.

3) Core Test Wrapper: The core test wrapper forms the interface between the embedded core and its environment. It connects the embedded core to the rest of the IC and also to the TAM.

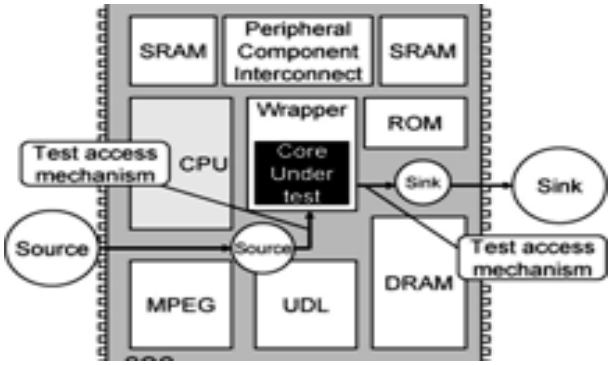

Figure 5. INTEGRATION OF CORES USING P1500 WRAPPER.

VI . SOC VERIFICATION

The verification challenges for SOC largely parallel those facing design and test V the challenge is the unprecedented complexity, and the hoped-for solution is through reuse V but the verification problem is in some respects harder. As the ITRS has noted [13], while design sizes have grown exponentially over time in accordance with Moore's Law, theoretical verification complexity has been growing double-exponentially, because the number of states that must, in theory, be verified is exponential in the size of the design. For example, consider an SOC built from n IP cores, with the ith core having some measure of verification complexity (e.g., number of reachable states, number of functional coverage points) of v(i). The goal is to try to find a way to reuse the verification effort for each core, so that the cores can be verified once in isolation. Doing so would reduce the verification effort to

Xn
vðiÞ þ vsys
i¼1

VII. SUMMARY

This paper provided a broad perspective on the reuse and integration issues associated with mixed-signal SOC design. Reusable forms of digital, analog/mixed signal, and programmable IP component were described. The platform-based design concept was illustrated using a Bluetooth baseband processor. Integration issues associated with interconnect, testing, and verification were presented. The authors believe that almost all designs in the future will make use of reusable IP and that commercial tool vendors will continue to advance their tools to address the more challenging issues of system level hardware/software co-design and co-verification.

ACKNOWLEDGEMENT

The authors wish to thank their students and research engineers for the significant contributions to the work at the System-on-Chip Research Laboratory at the University of British Columbia. The authors also wish to thank the Canadian Microelectronics Corporation (CMC) for licensing the infrastructure used in this work, and for invaluable discussions regarding SOC design and reuse.

REFERENCES

[1] HUANG,J.R.,IYER,M.K.,CHENG,K.T.:'A self-test methodology for IP core bus based programmable SoC's, VLSI Test symposium,19th proceedings on VTS 2001,2001pp.198-203.
[2] WILTON, S.J.E.,SALEH,R .:'Programmable logic IP cores in SOC design :opportunities and challenges' ,Custom Integrated Circuits, 2001,IEEE Conference on 2001 pp. 63-66.
[3] KIM,K.W.,KWANG,H.B.,SHANBHAG, N., LIU,C.L.,KANG S.M.:'Coupling-driven signal encoding scheme for low-power interface design',Computer Aided Design,2000ICCAD-2000.IEEE/ACM International Conference on,2000, pp. 318-321.
[4] YOO,S.J.;NICOLESCU,G.,LYONNARD,D.,BAGHDADI, A.,JERRAYA,A.A.,:'A generic wrapper architecture for multi-processor SOC co-simulation and design', Hardware/Softerware Codesign, 2001.CODES 2001.Proceedings of the Ninth International Symposium on,2001 pp. 195-200.
[5] GAISLER J.:'The LION Processor User's Manual',Version 2.3.7, August 2001, Gaisler Research,www.gaisler.com.
[6] AMBA specification,Revision 2.0,May 1999,ARM Ltd.,www.arm,com/Pro+Peripherals/AMBA
[7] HELLMICH, H.H.H.,ERDOGAN, A.T., ARSLAN, T. 'Re-Usable Low Power DSP IP embedded in an ARM based SOCArchitecture',IEE Coloquim on Intelectual Property, Edinburgh, UK,July 2000, pp. 10/-10/5.
[8] YEUNG, C.,MATTHEWS, G.,MORRIS, J.,HAVERINEN, A.,ZAIDI, J.:'Standard bus vs. bus wrapper: what is the best solution for future SOC integration?', Design,Automation and Test in Europe,2001.Conference and Exhibition

2001. Proceedings,2001pp.776-776.
[9] OELMANN,B.,O'NILS,M.:'Asynchronous control of low-power gated-clock finite-state-machines', Electronics,Circuits and Systems,1999.Proceedings of ICECS'99.The 6th IEEE International Conference on,Volume:2 pp.915-918.
[10] WU,Q.,PEDRAM, M., WU, X.W.:'Clock-gating and its application to low power design of sequential circuits',Circuits and Systems I:Fundamental Theory and Applications,IEEE Transactions on,Volume:47 Issuse:3,March 2000,pp.415-420.
[11] ERDOGAN, A.T.,ARSLAN, T.: 'Low power multiplication scheme for FIR filter implementation on single multiplier CMOS DSP processers',Electronics Letters, Volume:32 Issue:21,10 Oct.1996,pp.1959-1960.
[12] ERDOGAN, A.T.,ARSLAN,T.,HORROCKS,D.H.:'Low power multiplication schemes for single multiplier CMOS based FIR digital filter implementations',Circuits and Systems,1997.ISCAS'97.,Proceedings of 1997 IEEE International Symposium on, Volume 3, 1997 pp.1940-1943 vol.3.
[13] International Techbology Roadmap for Semiconductors. [Online]. Available: http://www.public.itrs.net
[14] M. Keating and P. Bricaud, Reuse Methodol-ogy Manual: For System-on-a-Chip Designs, 3rd ed. Boston, MA: Kluwer, 2002.
[15] R. Rajsuman, System-on-a-Chip Design and Test. Boston, MA: Kluwer, 2000.
[16] H. Chang, L. Cooke, M. Hunt, G. Martin, McNelly, and L. Todd, Surviving the SOC Revolution: A Guide to Platform-Based Design. Boston, MA: Kluwer, 1999.
[17] AMBA Specification (Rev 2.0), ARM Ltd., May 13, 1999.
[18] XTENSA Architecture and Performance, white paper, Tensilica Inc., sep.2002.
[19] The Mathworks.[online].Available:http:// www.mathworks.com.
[20] K.KUNDERT, H.CHANG, D.JETTRIES, G.LAMANT, E.MALAVASI,and F.SENDIG ,"Design of mixed – signal systems-on-a-chip," IEEE Trans. Comput-Aided Design Integer.Circuits syst.,vol.19, no.12, pp. 1561-1571, Dcl.2000.
[21] Y.C.JU,V.RAO, and R.SALEH,"Consistency checking and optimization of macro-models,"IEEE Trans.Comput-Aided Design Integer.Circuits syst.,vol.10,no,8,pp.957-967,Aug.1991.
[22] eASIC.[online].Available:http://www.easic.com.
[23] Faraday.[online]. Available: htt://www.faraday-tech.com.
[24] Stratix 2nd device handbook,Altera Corp., May 2005,datasheet.
[25] System.[online].Available : http://www.systemc.org.
[26] System verilog. [online].Available :http://www.systemverilog.org.
[27] P.Magarshack and P.G.Pavlin,"system-on-chip beyond the nanometer wall,"in proc.Design Automation Conf.,2003,pp.419-424.