System Modeling Services

Hemendra Talesara,
Verification Technologist
(htalesara@synapse-da.com)

Introduction

Transaction Level Models can provide a fast and more abstract representation of hardware. Virtual Platforms for early evaluation of systems can be developed using transaction level models. Synapse Design provides modeling services to develop virtual platforms to enable early development of software and hardware.

Trends in SoC

Synapse Design is currently working with many Tier 1 customers who are developing mega-million gate SOCs. In earlier days it was sufficient to do the spreadsheet analysis to get the architecture correct. However, lately it has become very difficult just to rely on spreadsheets or experience. The explosion of parameters that one has to consider to define the architecture are huge. Broadly considered, each of the following categories is a driver of SOC design today.

1. Multiple Cores
2. High Software Content
3. Low Power
4. Cost Sensitivity
5. Shrinking Market Window
6. Off-the-shelf Components
7. Mixed Signal Design

Design Implications

It is becoming increasingly difficult to use the traditional ways to analyze architecture. Let's look at some of the design implications for each of the design drivers identified above.

Multiple Cores

Multiple cores imply that there are multiple threads in the design. In earlier days, a single core design was easy to analyze for critical paths. Now, the traffic pattern of these simultaneously executing cores will affect the performance. Traffic patterns must also reflect the end use application. A Virtual Platform can help, if we are able to design a platform that can run these workloads at a significant simulation throughput to help with analysis. Using TLM2.0, it is possible to create models at the right level of abstraction that are register/transaction accurate, and with an acceptable level of simulation performance.

High Software Content

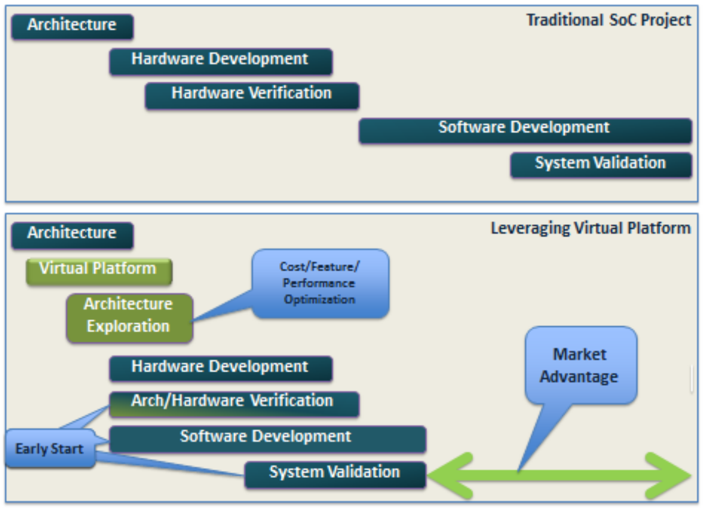
With ever-increasing software content, software is often in the critical path of the validation and power requirements of the SOC. Running software early during architecture tradeoff is needed to get the architecture right. An FPGA arrives too late in the design cycle to make reasonable tradeoffs. With a Virtual Platform it is possible to do this debug and development much earlier in the design cycle.

Low Power

For all mobile devices, battery life is a key differentiator and the architecture must support low power for a product to be viable. Software must be optimized early. Increasingly, hardware/software tradeoffs are made with power in mind. Some of these tradeoffs can be made at the architectural stage itself, preventing a late cycle re-spin.

Cost Sensitivity

With increasing competition and low cost products, profit margins have shrunk significantly. Competition is also requiring more and more features, but at a lower cost. This has clear implications for design. One cannot overdesign a product, as it increases the cost and cuts into the margin. One also cannot under design a product. If the desired features are not supported, you do not have



a viable product. Virtual Platform simulation can enable designing the optimal cost product.

Shrinking Market Window

Not just profit margins are shrinking, market window is shrinking too. Virtual gives an opportunity for early start on multiple fronts - verification, software development and system validation. Market advantage gained is well worth the additional effort required in early stages of development. Specially, since this effort is leveraged on multiple derivative products.

Off-the-shelf Components

New SOCs contain many off the shelf components, and there isn't a competitive advantage in re-designing many standard peripherals and buses. Most companies will instead focus on their proprietary IP for RTL design, where off-the-shelf components are not available. This has two implications.

One, the off-the-shelf components need to be highly configurable. Vendors are creating these components to meet a very large set of customers' requirements. That means when they design these components, they must test for a large number of configurations. These IP components can then be used by end customers in their SOC.

Second, it is in the vendor's interest to provide a virtual model that is highly configurable for testing purposes to their end customers. This is increasingly happening. Partly this is because customers are requiring it. Partly it is because such models were needed by the vendor themselves and therefore were already available. On the customer side, this really helps with SOC architecture analysis.

Mixed Signal Design

Finally, SOCs also contain some mixed signal design components. It is possible to create a Virtual Model that includes mixed signal components at a certain level of abstraction. This will allow simulation for complete system model validation, at a very early stage.

System Modeling

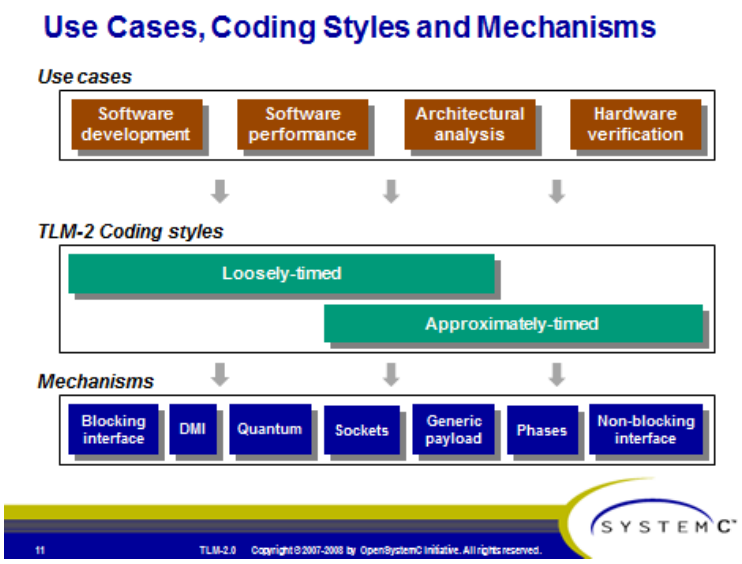
So, what does SystemC/TLM2.0 modeling offer?

There are three dimensions of any system that we want to capture, explore and evaluate - (1) function, (2) architecture and (3) performance. A HW/SW platform can be modeled in SystemC/TLM2.0 to enable this analysis. It's a platform on which we can develop and debug software. It's a platform on which we can do architectural exploration. It can have very fast simulation throughput without compromising architectural accuracy. It's a platform that can be leveraged to do the necessary functional verification.

Use Cases, Coding Styles and Mechanisms

If we adhere to coding guidelines, it is possible to create a model that executes at many orders of magnitude faster than the corresponding RTL model. Virtual Platform throughput does not depend very much on the number of peripherals. It's the peripherals that are in the critical path that count. Also, TLM2.0 allows you to create models at different abstraction levels. It's important to get the methodology right, if we want to maximize our return on investment.

Models for a virtual platform can be Un-Timed (UT), Loosely Timed (LT), Approximately Timed (AT) or Cycle Accurate (CA). We are generally familiar with Instruction Set Simulator as an example of UT model; or RTL as an example of CA model. In addition TLM2.0 introduces the concept of LT and AT model.



LT model has minimal timing detail needed to boot the OS or run a multicore system. The concept of temporal decoupling allows processes to run ahead of simulation time. This speedup is very helpful for some end application software developments. A key aspect of a modeling methodology is – "abstract away what you do not need"; such as timing that is only required for implementation. There is also a concept of DMI (Direct Memory Interface) that allows you to bypass the interconnect model to enable even higher simulation throughput.

AT models approximate cycle count accurately. Here processes do run in lock step with simulation time. These run slower than LT, but still much faster than RTL.

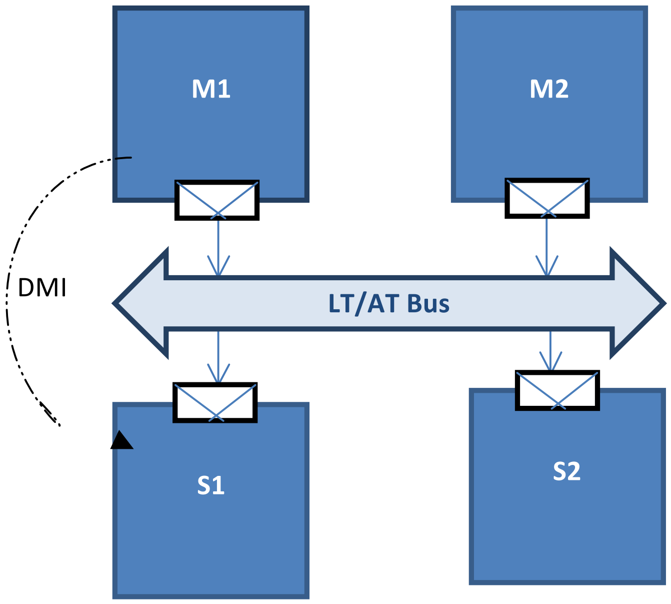
Another consideration that is important is the memory storage scheme. In the RTL cycle accurate world all data is copied and moved. However, at the higher level of abstraction, moving large chunks of data can be accomplished by a mechanism that uses references to memory and achieves much higher efficiency. Depending on the interface scheme picked, different strategies and implementations are adopted.

The TLM2.0 Class library supports some core interfaces such as blocking, non-blocking, DMI and debug transport. The core interface, along with a generic payload structure and base protocol, provides maximal interoperability for memory-mapped buses. The TLM2.0 standard thus serves as the glue to create a reusable, plug-and-play virtual platform.

It is certainly possible to design a modular model so as to add or enable the details as the use case changes. Modeling in such a way that communication and timing are kept separate from function does take a little longer to implement; but results in a highly reusable model with minimal changes.

Virtual Platform

A typical SOC will consist of many masters (cores), slaves (memories/peripherals); one or more interconnect / buses, plus other components. Buses are always in the critical path of delivering data. In TLM2.0 a bus can be LT or AT depending on the speed and accuracy desired. Depending on the use case, an abstraction model for the bus is picked. As mentioned earlier, for much higher speed (~10,000 x RTL) the DMI mechanism can be used. It is possible to boot the OS and develop applications on such a high speed system.



A typical Virtual Platform can contain some components as LT models, others as AT. It can contain some components that are cycle accurate RTL models, or pre-existing legacy untimed C/C++ models, or an instruction set simulator. These models can be connected through the appropriate design of TLM2.0 wrappers and adapters.

Huge Impact on Verification

Verification issues (testbench creation, random test generation and functional coverage etc.) have been well addressed in recent years using the System Verilog methodology. These testbenches often use UVM or such models for testing. They are created during the RTL phase of the design cycle. It is possible to leverage SystemC models that are developed much earlier in the design cycle as a test harness and combine with a UVM-like methodology. This reuse of SystemC models has a huge impact on both quality and schedule of verification.

These SystemC Models have been tested in the software development phase. Much of the test harness is already available as RTL design begins. Each RTL model as it becomes available can be verified in a more complete pre-tested SystemC environment. For generating tests, both new UMV tests and pre-existing software can be used. Coverage can still be implemented in UVM. In fact, the complete UVM testbench can be developed around the Virtual Platform, long before the RTL is ready. This gain is significant. This approach is useful for both component level verification and full SOC level verification. Integration issues are addressed up-front. Simulation throughput is certainly decided by the slowest link, in this case RTL. However, in validation phase, some customers replace critical parts of RTL with an Emulation engine and do system level integration and verification testing using top level software.

The SystemC models have a long shelf life. Many of our customers use them on multiple projects. They are used by multiple teams for different use cases. Once created, they become part of the customer's library. They are independent of the tools that may be used in the future. In addition to being compliant with TLM2.0 standards, guidelines and methodology to create modular and reusable components are needed. We work alongside our customer's design teams to create models, methodology and best practices that work for each customer.

Accellera Systems Initiative Notes

Accellera and OSCI merged about two years back. There is new standard on SystemC-AMS (Analog Mixed Signal) that is now available. UVM now has a TLM2.0 interface, allowing easy integration of UVM testbench with TLM2.0 Virtual Platforms. IP-XACT provides a standard structure for packaging and integrating IP. OCP-IP is also now under this initiative.

Summary

Current trends in SOC design have put new requirements on architectural analysis and software development. SystemC/TLM2.0 Standard allows us to create plug and play abstract models. Virtual Platforms using these abstract models allows optimization of cost, performance and features. Early start of software development and verification helps shrinks the overall schedule significantly. These models and virtual platforms, when well designed, are leveraged on multiple products.

Modeling Services Offered by Synapse Design

Synapse Design is supporting its customers with the transition to the new methodology, specifically in the following areas:

- *TLM2.0 Compliant modeling at various abstraction levels to support different requirements*
- *TLM2.0 Custom Library Element Development*
- *Methodology, Guidelines and Templates for TLM2.0 modeling*
- *Virtual Platform Development using mixed abstraction layers to support:*
    - *Early Software/Firmware Development*

  - *Architecture Exploration*
  - *Performance Analysis*
  - *HW-SW Co-Design/Co-Verification/Integration*
  - *Hardware Verification (In conjunction with SystemVerilog/UVM based testbenches)*
- *Methodology and Guidelines to support your team through development and deployment of a Virtual platform to support all use cases*
- *Test and Workload development to support analysis for the various use cases*