

Open Core Protocol: The SystemC Models

The System Level Design Working Group produces a set of SystemC transactional models for the OCP 2.0 specification

Open Core Protocol - International Partnership (OCP-IP) is a ground-breaking industry association, which provides a common standard for intellectual property (IP) core interfaces, or sockets, to facilitate plug-and-play system-on-chip (SoC) design.

The Open Core Protocol (OCP) is a common standard for the interfaces that connect IP in a SoC design. Since an SoC design may be built with IP cores from several different development teams and different providers with different architectures, there is a need for a common, open protocol that ensures the different components will work together on the chip. The OCP provides this standard, allowing plug-and-play connections of IP from multiple platforms. As an organization, OCP-IP guides the development of the industry standard by providing the specification, as well as EDA tools and models, to further ease the task of SoC development.

Before an SoC design can be manufactured, it must first be designed, modeled, and verified. SystemC, a modeling language and simulation environment, is an excellent choice for building a simulation model of an SoC. Just as the hardware is designed by choosing IP blocks and then connecting them together with interfaces, the simulation model is built from behavioral models of the IP blocks that are then connected together by channel models. (See figure)

CHANNEL MODEL LIBRARY

To aid in the design of SoC simulation models, the OCP System Level Design Working Group has created and released a set of OCP channel models for SystemC. The work was spearheaded by Nokia, in collaboration with Prosilog SA, Sonics Inc., and Synopsys, Inc.

These channel models provide a standard, point-to-point OCP connection in the SystemC simulation environment by standardizing the channel connection at several abstraction levels. Again, just as the OCP interface standard ensures that



Figure: The channel model handles the communications interactions between IP blocks.

hardware IP cores will work together on an SoC, the OCP SystemC channel models allow IP core behavioral models to communicate with each other during the simulation run. These standard channel models will lead to more simulation-model code interconnectivity and reuse. Because the channel models are already written and available, the simulation builder is free to concentrate on the custom IP models that make up the SoC simulation.

The OCP SystemC channel model library is made up of three different channel models: the OCP-TL1, the OCP-TL2, and the generic channel models. The OCP-TL1 channel is designed for simulations that are close to the hardware level where timing accuracy is required. The OCP-TL2 channel is designed for transaction-level simulations where higher speed and throughput are needed. Finally, the generic channel is built for user customization when a flexible interface is desired. Each of these channel models is described in detail below. The different channel models allow the simulation designer to choose the channel that best fits the models being connected.

THE OCP-TL1 CHANNEL MODEL

The OCP-TL1 channel model is the lowest level of the OCP channel models. This transfer layer channel is designed to connect SystemC models that are close to hardware models. The OCP-TL1 channel is close to cycle accurate and supports the phases, timing, and configuration parameters of the hardware specification. The OCP-TL1 model supports both blocking and non-blocking transfers and is event driven to allow the cores that are connected to proceed in parallel. Using this model, a master core can send a request, get the response, and accept the response all in the same cycle. Similarly, the slave model can get and accept a request and send a response (if required). In addition to providing correct timing, the OCP-TL1 channel also models all of the OCP signals, including the side band and control signals. All this makes this channel model ideal for connecting hardware models or other low-level models together.

The OCP-TL1 SystemC model has an application interface (API) especially designed to support all of the OCP 2.0 functionality. This includes burst transfers, multi-threading, optional data-handshake, busy threads, and pipelined requests, and responses. In addition, the API for this channel model

has commands to read and set all of the sideband and control signals. The sideband and control signals also have SystemC events associated with them. A core model, which is interested in an OCP signal, can wait for an event on that signal or it can poll the channel for changes.

In order to enhance ease-of-use of the model, blocking calls are provided which automatically handle some of the lower-level details of the OCP connection for the user. This makes it easier to connect basic models such as a simple slave. For example, the following function,

```
getOCPRequestBlocking( myRequest, true );
```

will wait until a new request has arrived on the channel, will store the new request into "myRequest", and then will automatically accept the request. These blocking calls may be used together with the model's non-blocking calls to suit the needs of the core model developer.

Because the OCP-TL1 channel model supports all of the OCP timing and signals, it may be used to verify that the core models connected to it are communicating correctly. The channel may be set to output a monitor log file that tracks the activity and state of the channel at each clock cycle. This log can then be automatically processed to verify that the models that were connected to the OCP-TL1 channel model are properly following the communications protocol. Runtime checkers are also built into the channel model to ensure that requests and responses are sent in the proper order.

THE OCP-TL2 CHANNEL MODEL

The second OCP channel model, OCP-TL2, is a transaction-level model with some timing information. This model is designed to provide a programmer's view of the OCP channel, where data transfer and throughput are the important factors. This model provides both higher speed and faster data transfers than the OCP-TL1 model, although with some loss of low-level timing accuracy. This model is meant to connect IP models that simulate at a higher level. The OCP-TL2 channel model may be used to verify the data flow through the channel and has full support for pipelined requests and responses.

The OCP-TL2 channel also uses an OCP-specific API which follows the same format and uses many of the same commands as the OCP-TL1 channel API and the OCP-TL2 channel also has full support for the OCP sideband and control signals. Whereas the lower-level OCP-TL1 channel model can only send one data word at a time, the OCP-TL2 model can send an entire transaction's worth of data at once. For example, the

following slave command sends a complete response of 32 words of data to the master:

```
sendOCPResponseBlocking( myResponse, 32 );
```

The command will block until the response has reached the master core and the master has accepted the response. In order to increase simulation speed, only a pointer to the data is sent over the connection.

The OCP-TL2 channel model also features serialized transactions, which make it easy to build test benches and to perform higher-level transactions. A single OCP-TL2 command can handle an entire transfer:

```
OCPReadTransfer( myRequest, myResponse, 32 );
```

This master command blocks until the 32 word read request has been accepted by the slave, then continues to block until the slave sends back a response which it then stores in "myResponse."

THE GENERIC OCP CHANNEL MODEL

The generic communications model is user customizable and may be modified to support any type of the user defined data class. This allows the generic model to connect to a wide range of existing IP interfaces. Just as with the OCP specific channel models, there is a lower-level, TL1 generic channel model and a higher, transaction-level, TL2 generic channel model. The OCP specific channels (OCP-TL1 and OCP-TL2) are built on top of the generic models.

The different OCP channel models allow the SystemC model builder to find the channel model that best matches the core models it connects. Since one channel model may connect two different types of cores, there is the chance that one core model would work better with a higher-level connection while the other model would prefer a lower-level connection. This problem is solved with OCP channel model layer adapters, which are able to convert commands from one API to another while handling any required buffering or timing issues.

MORE INFORMATION

The OCP SystemC channel models and full documentation are available to members and non-members alike through the OCP-IP website at <http://www.ocpip.org/socket/systemc>. ♦

Alan Kamas, www.kamas.com, is an independent consultant based in Sunnyvale, CA since 1995. He is an OCP Participant member and a contributing member of the OCP System Level Design Working Group.