Design Choice: Hard or Soft Virtual Components?

by

Thomas Harms
Marketing & Applications Manager, Europe
System-on-a-Chip Design Technology (SoCDT) Organization
Motorola Semiconductor Product Sector

To obtain a competitive advantage and secure increased market share, manufacturers are shortening design cycle times. As a result, cycle times in such hot electronic products as cell phones, digital cameras, DVD players, and internet routers have shortened to as little as three to six months. To meet these tight design cycles, designers require new methods and concepts in the chip development process.

The consumerization of most electronic products has pushed designers into shorter design cycle times and consumer expectations of expanded feature sets at ever lower costs. Continual advances in process technologies per Moore's Law enable designers to integrate more and more functions onto a single chip and utilize more transistors in a smaller space. With increased integration, new products emerge that feature smaller form factors, lower power consumption, and consequently, longer battery life.

However, designers simply don't have the resources available to develop new autonomous product designs from specification to tape-out in a few months long design cycle. The solution is to design a first-in-a-series platform for a new generation product offering. The design team can then leverage this platform to create future derivative versions of the product in rapid cycle times.

Facing a short design cycle, designers have to reuse as much as 90% of the original embedded blocks as is. The design team can use the remaining 10% of the blocks to incorporate new, advanced functionality. In cell phones, several manufacturers have added new phone platforms. Subsequent releases on the same platform have included new features such as extended battery life, web communications, and SoC designs for cellular phones which accept multiple standards.

Reusable Virtual Components (VC), also referred to as hardware or silicon Intellectual Property (IP), are available in one of two general forms. The first, soft VC, is delivered in synthesizable form and must be implemented into the target technology and verified by the derivative designer. The second, hard VC, is already fully implemented into a target technology. A hard VC block is delivered with much the same views as standard cell library elements, allowing the designer to rapidly integrate the VC into the derivative design. This paper will explore the differences between hard and soft VC and discuss the advantages and disadvantages of each from the viewpoint of both the VC author and the VC integrator.

Hard and Soft VC Defined

The Virtual Socket Interface Alliance (VSIA) defines hard VCs as follows:

> Hard VCs are optimized for power, size, or performance, and mapped to a specific technology. Examples include netlists fully placed, routed, and optimized for a specific technology library, a custom physical layout, or a combination of the two. Hard VC is process/vendor specific and generally expressed in GDSII format. It has the advantage of being more predictable, but consequently less flexible and portable due to process dependencies. Hard VC requires, at a minimum, a high-level behavior model, a test list, and full physical and timing models along with the GDSII data.

Soft VC, as defined by VSIA as follows:

> Soft VC is delivered in the form of synthesizable HDL code. The advantage is the flexibility of the source code so it can be retargeted to multiple manufacturing processes. The disadvantage is the difficulty in performance prediction (e.g., timing, area, and power).

The Semiconductor Reuse Standards (SRS) developed by Motorola's Semiconductor Product Sector (SPS) internally for our global, multi-site, multi-technology design environment follow these two definitions.

Integration in a Design

The table below (Table 1) compares Hard VC blocks to Soft VC blocks based on three different methods of implementation; reusability, retargetability, and configurability. By definition, a reusable VC block complies with the reuse standard's requirements for delivered views, the associated data formats, and view specific standards, such as coding standards. These views and formats are detailed, for example, in the VSIA VC Transfer specification or the SRS IP/VC Deliverables standard.

A VC block is retargetable if it is available in a high level representation, such as synthesizable HDL which allows the designer to easily migrate the VC block to new technologies or processes. Without a standard bus, retargeting to new architectures may also be required.

And configurability means that the VC block is parameterized to allow for adjustments by the user to meet specific functional or performance objectives. Examples of parameters include bus width, memory size, and enabling or disabling functional blocks within the VC.

	Reusable	Retargetable	Configurable
Definition	Available in the data views and formats defined by reuse standards.	At a high level of representation so that it can be rapidly be remapped to a different process or architecture.	Parameterizable to allow tailoring of functionality and performance.
Hard VC	Can be made fully reusable. Easiest way to integrate into SoC. Predictable performance.	Difficult to retarget and re-verify since reused on the physical level.	Not configurable.
Soft VC	Can be made fully reusable. Only performance goals.	Very easy to retarget to new process or technology.	Very well suited for configurability.

Authoring and Integrating Soft VC

To be reusable, a VC block must be available in the data views and the formats defined by the relevant reuse standard. This allows for easy transfer and, more importantly, efficient integration. Looking first at soft VC, the deliverables include a synthesizable HDL description which follows the Coding standards. Deliverables also include associated synthesis scripts to allow for synthesis, optimization, and scan insertion. The soft VC author(s) must also provide a verification testbench and an optional formal verification setup. This enables the VC integrator to verify that the gate level representation is functionally the same as the RTL description.

The soft VC author must also provide an extensive set of documentation to the VC integrator and SoC designer. These documents detail information about the VC's functionality, a description of the synthesis scripts, and how the scripts can be modified to new performance targets. The VC author must also create a functional verification plan describing the verification bench and the formal verification setup. And finally, the package must include a manufacturing related test strategy which provides all the necessary information to the integrator about how to test the VC in the embedded environment.

The system integrator must synthesize the soft VC to gates, perform scan insertion, and Automatic Test Pattern Generation (ATPG). The integrator or SoC designer must also place & route, back-annotate the timing, and verify the layout. Essentially the same design flow has to be performed on the soft VC block as for a real block of an SoC.

With its relatively small set of deliverables, the soft VC author can develop a block in a shorter time than hard VC. However, the implementation effort is now shifted to the VC integrator and SoC designer who need to perform synthesis, place & route, and functional and timing analysis as part of their product development cycle.

Authoring & Integrating Hard VC

Hard VC can be made fully reusable and is the easiest way for a designer to integrate a VC block into an SoC design for a given technology. Hard VC delivers highly predictable performance in terms of area, timing and power, and is ready to use in a plug-and-play fashion. Because hard VC has been modeled according to the SoC design system and target technology, it delivers confirmed and reliable performance, is proven in silicon, and offers the design team established quality. The hard VC author, however, has to expend the additional effort to provide complete documentation compliant to the relevant standards including all relevant views, netlists, timing analysis, verification, etc.

Summary

Electing reusable VC as a core design strategy is an easy call for design teams facing ever shorter design cycle times. To meet these shortened cycle times efficiently, design teams are developing first-in-series platforms and reusing VC. Product developers can leverage the original platform with rapid upgrades to the feature set by reusing as many as 90% of the existing blocks. They can produce the derivative by implementing new features with as few as 10% of the blocks. So reuse becomes the key enabler for rapid SoC design turnaround.

Reusable VC is available in one of two basic forms: soft or hard VC. Hard VC is ready to drop into the embedded SoC environment in a plug-and-play fashion. Reusable soft VC offers the integrator the ability to retarget a VC block for a new process or technology. Soft VC delivers only the source code and requires fewer deliverables (than hard VC), leaving the integrator responsible for "hardening" it into the specific application. In the ideal case, all VCs are being developed in a synthesizable soft VC form for easy migration to new technologies, hardened and qualified into a hard VC form by the VC author, and then shipped to the SoC designer to enable the required cycle time reduction.

References

The VSIA specifications are available at **http://www.vsia.com**.
The Motorola Semiconductor Reuse Standards (SRS) are available at **http://www.mot-sps.com/srs**.

* * * End * * *