Choosing an Intellectual Property Core

MIPS Technologies, Inc.
June 2002

One of the most important product development decisions facing SOC designers today is choosing an intellectual property (IP) core. It can impact product performance and quality, as well as time-to-market and profitability. But SOC designers face many challenges when choosing a core. Determining which core is most appropriate for a given SOC requires careful consideration. Decisions must be made about the type of core (soft vs. hard), the quality of the deliverables, and the reliability and commitment of the IP provider. This paper discusses each of these areas and offers guidance on how best to evaluate the features of competing IP cores.

Introduction

Continuing improvements in silicon manufacturing technology have made vast amounts of silicon real estate available to today's design engineers. Unfortunately, the ability of engineering teams to design circuits has not kept pace. This imbalance has spawned the IP core industry. IP cores allow design teams to rapidly create large system-on-a-chip designs (SOCs) by integrating pre-made blocks that do not require any design work or verification.

A number of difficult challenges accompany this new design style. Depending on the core, they can be minimized or exacerbated.

First of all, IP cores may be delivered to customers in one of two forms: soft or hard. In both cases, the customer receives a functionally verified design. A soft core, also known as a synthesizable core, is synthesized by the customer and implemented in its SOC. A hard core, on the other hand, is fully implemented and ready for manufacturing. (Technically, a design is not implemented until it is manufactured. In this context, however, implemented means laid-out and ready for manufacturing.) The SOC team need only drop the hard core into the chip as a single monolithic piece. Soft and hard cores have different problems and benefits, which are addressed below.

An IP core jump-starts a key part of the SOC design task. The design team gets a verified design, which enables them to complete their chip in less time with fewer engineering and EDA resources. However, integrating a core into a chip requires many steps. How easily this is accomplished, if at all, depends on the deliverables provided. This paper details some of the collateral deliverables that enable easy core integration into all stages of the SOC design process.

Finally, there is the IP vendor to consider. The IP industry is still young and there have been a number of poor products and even some failures, and they have not been confined to start-ups. Consequently, a customer must evaluate not only the IP core, but also the IP provider.

Soft vs. Hard Cores

Let's examine the pros and cons:

Performance
Because soft cores are not implemented, they are inherently more flexible in function and implementation than hard cores. On the other hand, hard core developers can afford to spend more time optimizing their implementations because they will be used in many designs. Thus, there is a perception that hard cores offer higher performance.

In fact, high-end, full-custom hard cores designed for the most advanced processes do offer more performance than soft cores. By using latches, dynamic logic, 3-state signals, custom memories, and so on, the full-custom design team can achieve much better results than a fully static synthesized design. For an SOC that requires performance that pushes the limits of current process and design technology, a full-custom hard core is better able to meet these needs.

However, if the performance target is within the range of a soft core, then the potential performance advantage of a hard core is immaterial. The SOC design team can meet its performance goals with a soft core while taking advantage of its inherent flexibility. (As process technology improves, the maximum frequency limits of soft cores will also improve, making them an option for even more SOC designs.)

Even at slower clock frequencies, a hard core may offer an advantage in terms of silicon area. But this is not always true. Often, a hard core is simply hardened using an ASIC-style methodology, which offers no advantage in area of speed. In other cases, a full-custom core is not re-optimized for each process generation, thus diminishing its frequency and area advantages.

Technology Independence & Portability
One of the advantages of a soft core is that it is technology independent. That is to say, the high-level Verilog or VHDL does not require the use of a specific process technology or standard cell library. This means that the same IP core can be used for multiple designs, or for future generations of the current design. (Some soft core IP providers use design styles that make their cores technology-dependent, but the advantages of this approach are unclear.)

A hard core, on the other hand, is very technology-specific. In fact, if a foundry changes its process parameters or library factors, a hard core may not work correctly with the process tweaks. This introduces risk since the IP provider will need to re-verify the hard core when process parameters are changed.

Hard cores can be ported to new process technologies, but the effort to re-optimize full-custom cores is significant and costly. It may take two years or more for some advanced microprocessor cores. Because of this, hard cores are often optically scaled for new processes. While simple and

fast, this technique diminishes many of the advantages of the full-custom optimizations done by the design team for the original process.

Furthermore, optical scaling introduces additional risk, since it only guarantees that the new design meets design rules. It does not guarantee correct timing or function. Since the optical scaling is a short-cut design style, it can be very difficult to fully re-verify an optically scaled IP core.

In reality, soft cores are likely designed with one technology and library in mind. The design itself is independent of this choice of technology but it optimized for this one technology and library. Similar technologies will be near-optimal, but some significantly different technologies (for instance, ones with very slow RAMs) may not see equivalent results. However, this effect is secondary. Soft cores will generally be better optimized than optically scaled hard cores.

Speed/Area/Power Optimization
Hard cores are optimized once, when they are implemented by the IP provider. Because the core is optimized only once, the IP provider can afford to spend significant resources. Thus, a hard core will typically run faster than a comparable soft core for that one technology in which it is implemented. But, even in that single technology, it is only optimized for one set of goals. If the goal is low area at reasonable performance, the highly tuned performance-optimized hard core may be too large for the application.

Soft cores, on the other hand, can be "application optimized": Timing, area and power targets can be adjusted to fit the specific embedded SOC design. For instance, if an SOC uses a 200-MHz clock, then a soft IP core that was designed to run at 250 MHz can be targeted to run at exactly 200 MHz instead. This allows for smaller area and lower power while still meeting the design constraints.

This application optimization also extends down to low-level IO timing. The IO constraints of a soft core can be adjusted to exactly fit the environment the core will be used in. If a hard core has a late output signal, there is little the SOC designer can do to improve that timing.

If an SOC's speed, area and power targets are exactly what the hard core was targeted for, then that hard core will be competitive. For the great majority of designs, however, a soft core will be better optimized for that particular SOC.

Customizability
Soft cores offer another advantage over hard cores: compile-time customizations. These are design options chosen prior to implementation.

Cache memory size is a common compile-time customization. A soft-core processor can be configured for exactly the cache size needed by the specific embedded application. A hard core, on the other hand, cannot be customized in this way.

Another customization employed in many soft cores is instruction specialization, or optional support for certain instructions. For example, support for external coprocessors may be

optionally included if required by the SOC. Special hardware to enable instruction code compression may also be needed in some systems. However, in the systems that do not use these features, the extra hardware could be removed in a soft core, saving area and power.

Soft cores may also include implementation configuration parameters. These are a special kind of compile-time customization that helps the soft core better match the design style used by the SOC team. For instance, a microprocessor core is typically implemented using gated clock circuits. However, this type of clocking may not fit well with some clock routing tools. If the processor core offers a compile-time setting that changes all gated clocks into equivalent recirculating MUXes, this could be used by the SOC team to make implementation easier.

Ease of Integration

Unless a hard core has been implemented by an internal design group, a soft core is more likely to be easily integrated into the flow used by the SOC design team. The reason is that SOC design teams will be adding RTL modules around the IP cores they have licensed. The cores will then look just like the other modules of the SOC and can be implemented like them, too.

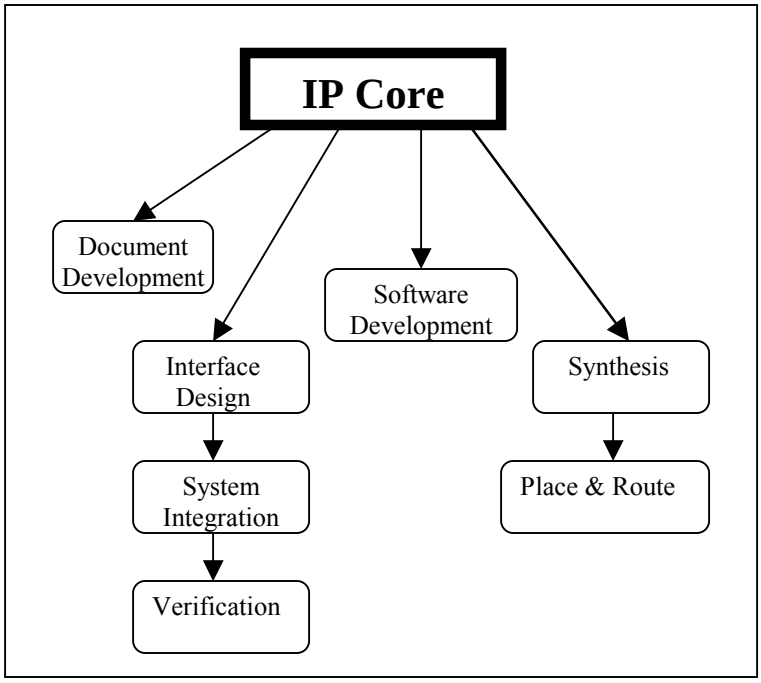
A hard core, on the other hand, will look more like a black-box RAM, especially if it was implemented using full-custom techniques. This means that a hard core provider will need to supply many more black-box models of the core so the SOC designers can design their modules around it. This is inherently more difficult than using a soft core. For instance, a full-custom hard core may not have a gate-level netlist. This is because the design has been done at the transistor level, and gates were not used. But, the design team may need to run gate-level functional simulations with back-annotated timing. This will be difficult due to the lack of a gate-level netlist.

Collateral Material

A competitive soft IP core is not just a collection of Verilog or VHDL source files. By the same token, a good hard core is not just a layout database. Today's IP cores consist of a suite of deliverables that allow the SOC design team to integrate the IP core into their design. The goal of these collateral deliverables is to make integration of the IP core into all parts of the design process as easy as possible.

Figure 1 shows the various SOC development activities that are affected by the IP core. This section discusses some of the deliverables that are necessary for both soft and hard cores.

Figure 1: Development tasks affected by IP cores



Documentation
Clear and concise documentation is a prerequisite for most technical products. However, the number of different people who need to refer to IP core documentation is large and varying, which makes the documentation for an IP core particularly challenging.

In Figure 1, each of the development activities has different documentation needs. For instance, software developers need to know the programmable features of the hardware, but they probably do not care to know how it is implemented. Thus, a good set of documentation makes it easy for software developers to find all of the information they need without wading through information they do not need.

Finally, if the SOC team will be creating documentation for their SOC that could reuse parts of the IP core documentation, the IP provider should supply editable documentation source files and the rights to include excerpts in the SOC documentation.

Interface Checkers
The SOC team must design logic that interfaces with the various signals and protocols of the IP core. To determine if it has been designed correctly, the IP provider can deliver interface checker modules to verify correct operation of all interface signals and protocols. It can be as simple as ensuring that static signals do not change, or as complex as validating the correct operation of a multi-cycle bus protocol.

These checkers can greatly simplify the effort required of the SOC team by automatically verifying correct operation for a given type of interface transaction. In the case of an illegal

transaction, the checker should report the error so that the SOC designer can easily pinpoint the defective logic and debug the failure.

Interface checkers must work correctly in the SOC design environment. They should be easily integrated into functional simulations, while not present in actual hardware.

Protocol Tabulators
The IP provider can supply another deliverable to make the interface verification easier: the protocol tabulator. This is a module that monitors the interface transactions, watching for various corner cases. The protocol tabulator remembers all of the types of transactions seen and reports the corner cases that have not been exercised. The IP provider must supply a list of corner cases that are required to enable full verification of the interface.

During development, the protocol tabulator will help the SOC team determine what corner cases remain to be verified. Once development is complete, it also gives the SOC team the security of knowing that they have exercised all of the necessary corner cases. Since the IP provider has the best understanding of the core interfaces, this list of corner cases will be much more complete than anything the SOC team could devise.

RAM Checkers
If an IP core has internal RAMs that the SOC team must compile and integrate, there is a chance that bugs will be introduced during the process. Debugging failures caused by deeply embedded RAMs is very difficult for the SOC team since it typically involves tracing failures through internal core modules. A RAM checker can greatly ease the debug of failures caused by the RAM models. By quickly recognizing failures at the RAM interface, the SOC team can avoid debugging the internals of the IP core and quickly resolve the problem internal to the RAM. (It is a bad situation when the SOC team has to debug through an IP core. They should be able to rely on its correct operation.)

Fast Simulation Model
For SOC designers, simulating the full SOC with the RTL of a large IP core may be very slow. If the IP provider can supply a fast functional model of the core that is cycle accurate, the customer benefits from faster simulations, faster debug, and less usage of simulation licenses. Even a non-cycle accurate model may be good enough for most SOC design and debug. As long as a final run of the cycle-accurate model is made, a fast functional model is beneficial during development.

EDA Tool Support
Another indicator of the quality of a core is the breadth of EDA tool support. Since different design teams may use different tools, deliverables in multiple formats supporting many EDA tools are typically provided in today's advanced cores.

For instance, even if an IP core was designed using Verilog, VHDL is required by customers who have EDA tools and methodologies built around VHDL. If a core is only delivered in Verilog, then the SOC team will have to go through a cumbersome and bug-prone translation in order to use the core.

Furthermore, the IP provider should deliver more than the required format. Different EDA tools may have different implementations of standard formats. In the example above, an IP provider cannot supply only Verilog RTL to a Verilog customer; it must support the specific Verilog simulator to be used. Otherwise, the customer may end up debugging design problems related to a Verilog simulator that runs a little differently than that used by the IP provider.

This concept can be generalized to virtually all deliverables. For hard cores, this concept is also applicable in the implementation phase. A hard core must be delivered in a format that is supported by the SOC team's backend tools. And the IP provider must support the specific backend tools to be used.

Sample EDA Scripts

To help jump-start the various design activities, the IP provider should supply example scripts for supported EDA tools. This is yet another way the IP provider can enable the SOC team to efficiently bring up their system using the IP core. The scripts may be as simple as makefiles to enable a compiled functional simulator. They may be as complex as a complete suite of scripts designed to automate the execution of functional regressions. In any case, example scripts are almost always useful for SOC designers.

For soft cores, example synthesis scripts are almost a requirement. At a minimum, they should give top-level constraints, false-paths, and multi-cycle paths. If possible, scripts implementing several industry-standard synthesis methodologies should also be included. Of course, the simpler these example scripts are, the easier it will be for the SOC designer to understand, modify, and integrate into his or her flow.

Functional Core Verification

Although SOC designers do not change the RTL design of the soft IP core, they do change some functions as a normal part of chip design. Examples of things that change the function of the design include scan-chain insertion, clock buffering and RAM BIST integration. The SOC team needs to be able to verify that these changes have not affected the correct operation of the core.

One way to verify that the new design has not functionally changed from the old one is for the IP provider to supply a test bench and test suite to fully verify the correct operation of the core. Unfortunately, the complete test suite for many cores is too large to be delivered as part of the IP core. So, most IP providers select a subset of the full verification suite that can be run to verify correct operation. Most of the time, this subset is more than adequate to find any bugs that may have been introduced by the above types of design changes.

However, formal verification tools are a much more thorough method to guarantee correct operation. These tools mathematically prove that the new design is equivalent to the old one. Support for formal verification tools can eliminate the need for the SOC team to run the above gate-level regressions.

Software Co-Development Tools

The standard way to develop software for a new system is to first manufacture sample hardware

and then develop the software to run on it. In many cases, however, this prolongs the time to market, so software development often occurs in parallel with hardware development.

Software development requires much faster system simulation than hardware development. Thus, the IP provider must deliver a very fast functional model of the IP core. This provides enough performance for low-level firmware development.

For greater simulation speed, hardware logic emulators are sometimes used that can run an order of magnitude faster than a pure simulation (though they are still 2-3 orders of magnitude slower than actual hardware). These tools are notoriously difficult to use and require special synthesis. For SOC teams that plan to do hardware and software co-development, support for these technologies is a key requirement from the IP core.

Evaluating the IP Provider

There are many companies that offer IP cores. Many are small, start-up design houses, and some are large, well-established companies using IP cores as a new method of delivering their designs to customers. Unfortunately, the size of a company is not an indicator of IP core quality. The SOC designer should verify the commitment a company has made to IP core products.

Designed for Reuse?

For example, an IP provider that is not completely committed to IP cores has offerings that may only be previous designs repackaged as IP cores. A company that is serious about building high-quality cores will design them from scratch with reuse in mind. This section details some hallmarks of designs made to be reusable.

First of all, watch out for soft cores that are the source code for a full-custom hard core. Since these designs were not originally made to be synthesizable, they will be poor products when compared to those designed to be synthesizable. When making a hard core, optimizations can be made based on the known implementation style. However, in a soft core, the implementation is not yet done, so these shortcuts should not be taken since they may result in non-functional or sub-optimal implementations.

Another thing to look for in a soft core is registered interface signals. By registering IOs, internal logic can be timing-independent from anything the SOC team hooks up. Furthermore, it enables easy timing predictability and gives very good timing constraints to the SOC designer. All of these things make SOC design easier.

A soft core that was designed from the beginning to be reusable will typically have more configuration choices and more flexibility in implementation. It will also likely be delivered with multiple design environments in mind. A design made without reuse in mind will be less flexible in function and implementation.

Complete Product Line

Another sign of a good IP provider is a complete IP core product line. If you choose a soft core, make sure that the company offers a complete soft core product line for future product

enhancements. If you choose a hard core, make sure it is offered in all of the process technologies that you will be using.

Also, verify that the IP provider has a clear direction for future IP core development. Do they plan to expand their soft core offerings? How do they plan to port hard cores to new process generations? (Watch out for optical scaling of full-custom cores.)

Maintenance and Support
The quality of product maintenance and support is not unique to IP cores. However, watch out for young companies that lack dedicated support. Even for established companies, the infrastructure required for maintaining IP cores is somewhat specialized. Here is a checklist of items to look for:

- Does the company have a clearly documented way for the customer to get help in answering questions?
- How is the SOC team charged for support? (Are you in danger of losing support?)
- Is the company forthright in disclosing bugs in its designs?
- How often does the company make new releases to fix bugs?
- Does the IP provider make maintenance releases that add new functionality to the IP core or its collateral deliverables (e.g. support for more EDA tools)?
- How responsive is the company once a support issue is submitted?
- If the support response is too slow, can an issue be escalated?
- How knowledgeable is the first-line support staff?

In many cases, the quality of support is not a part of the initial IP core purchase decision. However, poor support can become a major problem at a time in the project when the design team desperately needs help. The highest quality support is imperative for project success.

Conclusion
The field of IP core design is a new one. There are many companies vying for design wins in this rapidly expanding field. The SOC designer must be careful to evaluate the designs and the IP provider companies carefully to avoid the many pitfalls that can happen with any new technology.

The optimizations used in creating a hard-core may be useful for a minority of designers whose requirements are exactly what the hard core was targeted for. But, the flexibilities that are possible with a soft core will be optimal for the great majority of designs:

- Application optimization
- Compile-time customizations
- Technology independence
- Ease of integration into SOC flow

An IP core with poor deliverables can also be difficult to integrate into an SOC flow. Therefore, it is very important to evaluate the IP core deliverables to make sure the correct EDA tools are supported and all steps of the SOC flow can be addressed properly.

The choice of the IP provider is perhaps as important as the choice of the IP core itself. An IP provider that is making a significant commitment to IP cores is an absolute necessity. Furthermore, the SOC team needs to know that the IP provider will be there in the future to support the product as well as to introduce the new products.

There are many challenges facing today's SOC designer. Using a high-quality IP core from a reputable company should make those challenges easier, not more difficult.