Searching the Ideal Core for FPGAs

With the exponential increase in density and performance of programmable logic, cores have become essential to further improve the time time-to-market benefits for FPGAs. Hence, FPGA vendors are striving to implement the "ideal core" solution for system-on-a-chip on high-density FPGAs.

The ideal core must have predictability as an essential part of its characteristic. Specifically, core performance must be met every time the design is compiled, independently of development tools, device size or adjacent logic.

In addition, the ideal core must deliver high-performance and minimal area to support new emerging applications such as 66 MHz PCI and high-speed DSP.

Finally, the ideal core must provide high-flexibility and low design risk. After all, flexibility is the fundamental value of programmable logic that has made it the fastest growing segment in the electronics industry for the past decade.

There are several avenues to achieve some of the characteristics of an ideal core, but often at the expense of sacrificing others. One method is to supply a "soft core" implemented in a standard FPGA. A soft core can be a synthesizable HDL design without optimization or pre-defined placement and routing. This type provides high flexibility, but at the cost of performance and predictability. By adding optimization for a specific FPGA architecture, and pre-define placement and routing, both performance and predictability of a soft core can be significantly improved. A second method is to supply a "hard core" embedded in silicon which solves the performance and predictability limitations of a soft core, but fails to deliver on flexibility.

In addition, generating the ideal core for FPGAs goes beyond an efficient core design. A core friendly FPGA architecture, smart core implementation tools, and system level solutions such as prototyping boards and test benches are equally as important as the core itself

Flexibility is the most important feature of programmable logic

The flexibility of a programmable device is especially valuable when implementing complex, high-performance cores like PCI, which must meet a strict but evolving specification. It is well known that PCI is difficult to do right the first time. The reasons are that the demanding high-performance PCI specification pushes the technology to the limit, each application requires its own unique configuration, and a PCI end product must work with many vendors' chip-sets and PC platforms. Because of the difficulties designing PCI, it is not uncommon that standard PCI chips are released with an errata list of known bugs and limitations.

A soft core solves this issue. For example, Xilinx released the first LogiCORE PCI interface in February 1996. It's a soft core with pre-defined placement and routing that the designer can customize for a targeted application via a graphical user interface. Since the first LogiCORE PCI product, five major revisions have been released to add new features, improve performance, and support new FPGA devices. As a result, the users have had access to a full-featured PCI interface that can be customized for a specific application and implemented in the latest generation of FPGA technology. Furthermore, the designer can benefit from future enhancements of the PCI core, to support new PC platforms and future revisions of the PCI specification. Even PCI boards that are already released and shipped to the end-user can be revised with a soft upgrade. Obviously, this degree of flexibility would be impossible with an embedded hard core.

Because a soft core can be used with a standard FPGA, the designer can choose from wide range of FPGA sizes, packages and speed grades to implement the core. As a result, the FPGA that gives the most cost-effective solution for a target application can be chosen. In the case of a hard core on the other hand, the specific function block is embedded in silicon, in a specific FPGA device. Typically, the number of applications for this specific FPGA is limited. To not loose the large-scale manufacturing benefits, the vendor is often forced to support only a few FPGA sizes and packages. Another reason for the vendor to limit the number of supported FPGAs is that, if a problem occurs in the hard core, all supported FPGAs

must be revised. The result is that many users have to choose a device for the application that is bigger, faster and more expensive than necessary.

Based on this discussion, functions that can be embedded but still provide flexibility are still good candidates for being embedded as hard cores. An example of such a function is RAM. RAM is used in almost every system, although, with different configuration and sizes. Because RAM in it self is programmable, it can be embedded as a hard core, and still be configured to fit a specific application. Where a few big blocks of RAM are required, an embedded hard core implementation can be beneficial. However, where several small blocks or RAM are required, such as in many DSP applications, a soft implementation is still most effective. To maintain the flexibility of the embedded RAM it is important that it is configurable. For example, it must support different depth and widths, synchronous/asynchronous read and write, and single/dual port configuration.

Another type of function that is suitable as a hard core is J-Tag. This is a commonly used function, that occupies minimal silicon area, hence, can be embedded in a complete FPGA family without negatively affecting the cost.

Embedded cores are smaller, but the advantage decreases on large FPGAs

There is no doubt that the programmability of an FPGA introduces a significant overhead in silicon area and that an embedded hard core therefore is more area efficient. However, optimizing the implementation for the targeted FPGA architecture can reduce the occupied area of a soft core. As a result, the advantage of a hard core becomes less notable. By pre-defining placement and routing, and by providing necessary constraints with the soft core, the designer can achieve this optimization without manual intervention.

Nevertheless, embedding a complex core function in silicon significantly reduces the required area. The amount of area that is saved for the whole FPGA varies depending on the type and the complexity of the core, as well as the gate count of the targeted FPGA. For example, Xilinx' programmable PCI Master interface core is equivalent to about 12,000 gates. Even though, it is highly optimized for the targeted FPGA architecture, an embedded version of the same core would save about 80% in area. This assumes using a gate array technology and takes into account the overhead required for connecting the PCI core to the I/Os and to the back-end FPGA array.

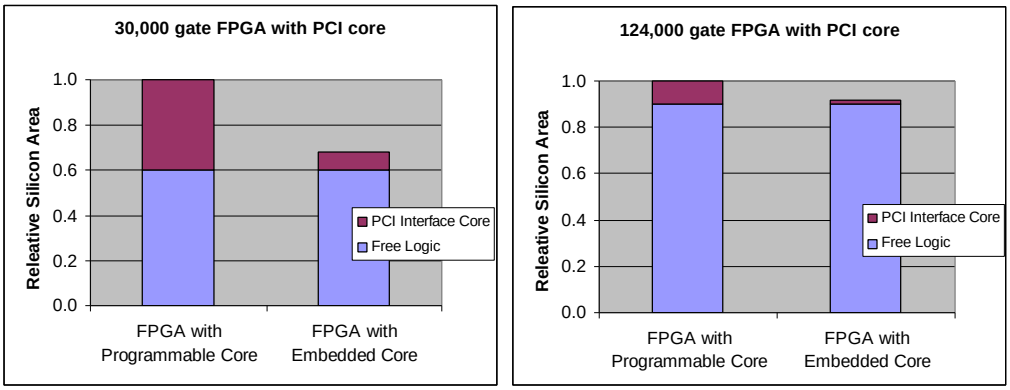

Figure 1: The difference in area for a soft core and a hard core declines with FPGA size

The 80% area reduction applies for the PCI core only - not for the complete FPGA. The area reduction for the complete FPGA declines with the FPGA size. When the LogiCORE PCI interface was introduced two years ago, it was optimized for the XC4013E, which was one of the larger FPGAs available at that time. This FPGA integrates typically 30,000 system gates.[1] Consequently, the soft implementation

[1] Assumes that 30% of the FPGA array is used for memory.

of the PCI interface occupies about 40% of the chip. If the core would be embedded, it would be 80% smaller, but the die size of the whole FPGA would only be 32% smaller.

A year later, the same core was optimized for the XC4062XLT that integrates about 124,000 system gates. The soft implementation of the PCI interface now occupies less then 10% of the FPGA. In this case, embedding the core in silicon would only result in an 8% smaller die. Figure 1 illustrates the difference in area between a soft core and a hard core in a 30,000 gate FPGA and a 125,000 gate FPGA. FPGA vendors today ship 250,000 system-gate FPGAs and are rapidly approaching 1 million gates, hence, the positive effect on silicon area for the same embedded core will become even less.

As seen, the area saving for a 10-15K gate embedded core such as PCI becomes insignificant in bigger FPGAs. As FPGAs grow only very complex functions will be beneficial to embed as hard cores. Yet, more complex functions typically require more flexibility and support fewer applications, which negatively affects the fundamental benefits of an FPGA, as discussed earlier. Again, a core that does meet the requirements for a hard core is big blocks of RAM.

How does smaller silicon area affect the component price?

Die size is not really important to the end-user, unless it results in a lower component cost. To understand what effect a smaller die will have on the cost, we must consider all factors of the total cost. As shown in Figure 2, the total cost to manufacture an FPGA includes not only silicon, but also assembly and test. In fact, the silicon cost is decreasing relatively to the total cost with more advanced process technologies and denser chips. For example, in a 0.5µ technology the silicon cost represents less than 40% of total cost. Consequently, in the PCI example described above, a reduction in silicon area by 8-32% will only save 4-12% on the total cost.

Moreover, the cost for manufacturing test increases by about 10% due to the complexity of the embedded core, which offsets out the difference even more.

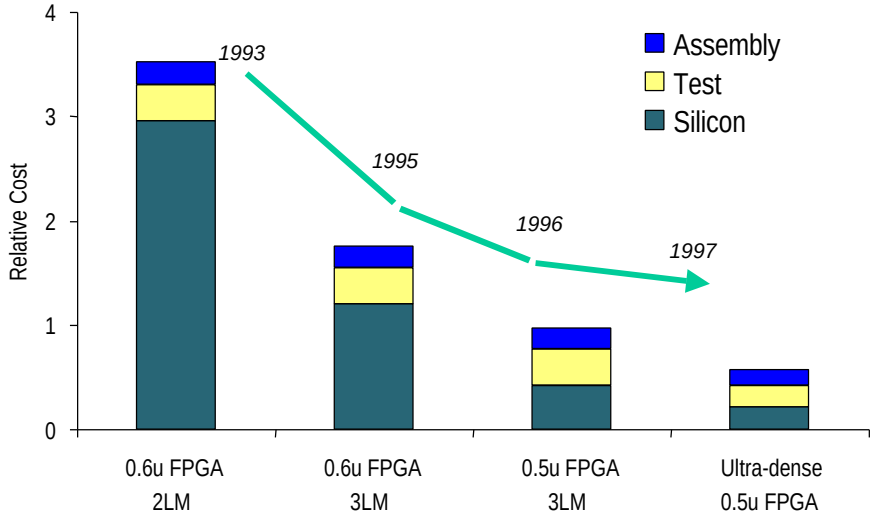

Figure 2: Silicon cost becomes a smaller component of total cost with denser processes

Typically for the semiconductor industry, manufacturing volume significantly impacts the component cost. At a high volume where the fixed NRE can be amortized over a large number of devices, the unit cost obviously goes down. The programmable logic market greatly benefits from this principle. In oppose to the ASIC market, each FPGA customer typically buys in moderate to low quantities. However, because the number of programmable logic customers greatly exceeds the number of ASIC customers, and

the same FPGA is used in multiple applications, the FPGA vendor can benefit from large-scale manufacturing.

When an FPGA becomes too specialized, the up front NRE cost, including development, mask-sets, engineering wafers, and software tools, greatly affects the unit price negatively. In fact, if the up front NRE cost is added in the PCI example, the 4-12% saving in component manufacturing cost is eliminated and the embedded version of the core becomes more expensive than the soft core.

Moreover, because a specialized FPGA must be treated as a separate product, with a unique part number, the customer must account for an incremental logistical cost. This cost includes qualification, handling and inventory.

Only commonly used functions that benefit a big enough segment of the market make sense to implement as hard cores. Ultimately, if an automatic ASIC migration path is provided, the completed FPGA can be cost reduced for high-volume production after being fully verified and tested.

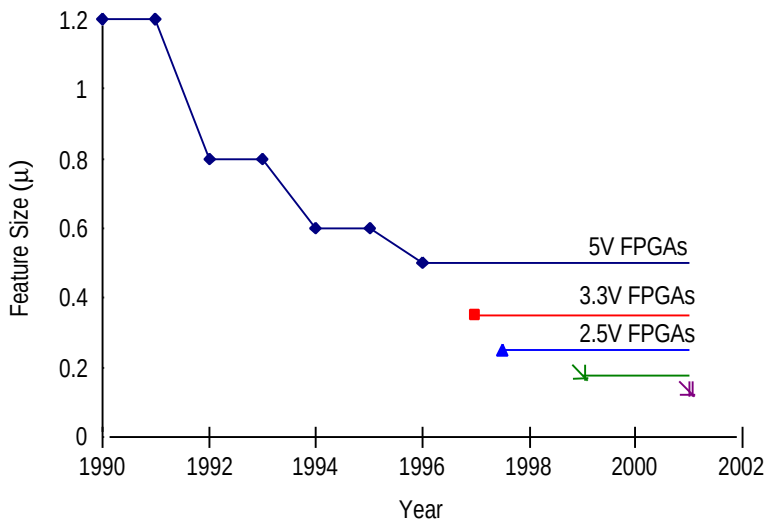

Figure 3: Accelerated FPGA process migration enabled by a structural repetitive architecture

An embedded hard core will lag in process technology

Perhaps the strongest argument for an embedded hard core is higher performance. By embedding an optimized hard core in an FPGA, using gate array or standard cell technology, the overhead of a programmable architecture is eliminated with higher performance as a result. An FPGA with embedded hard cores could then be used in super high-speed applications that previously only have been addressed by ASICs. Examples of such applications are RAM-Bus and G-bit Ethernet.

However, considering the longer development time for an embedded hard core, and the accelerating process migration of standard FPGAs, the performance gap between hard cores and soft cores is closing. As shown in Figure 3, the FPGA industry has over the past 5 years aggressively moved to new generations of process technologies, from 1.2µ in 1992, to 0.25µ in 1997. As a result, vendors have been able to drive the FPGA cost down, and, even more importantly, the integration capacity and performance have significantly gone up.

The fast adoption of new technology is possible because the structured, repetitive FPGA architecture is relatively easy to migrate from one technology to another. A highly specialized embedded core function, on the other hand, is more complicated to migrate, and requires significant time for redesign and re-verification. As a result, a specialized FPGA with an embedded core would be 6-18 months behind

the standard FPGA. In other words, a specialized FPGA would be one process generation behind the standard FPGA.

When comparing the performance of the two types of cores, this lag in process technology must be taken into account. Using PCI as an example again, the Xilinx XC4000XLT family, implemented in a 0.35µ process, is currently fast enough to support a soft implementation of a fully compliant 33MHz PCI interface. The next generation FPGAs are targeted for a 0.25µ process, which combined with a refined architecture, will more than double the performance. It is estimated that this new family will support 66MHz PCI implemented as a soft core by the end of 1998. This is about the same time when a 66MHz PCI interface implemented as an embedded hard core would be released.

The performance gap between soft cores and embedded hard cores can be reduced even more by careful optimization of the soft core to the targeted FPGA architecture. Signal delays in an FPGA is largely determined by interconnects, hence, careful floorplanning and optimized routing can significantly improve the performance. As an example, benchmarks show that designing and optimizing an 8-bit, 16-TAP FIR filter to utilize available architectural features such as look-up tables and registers, can result in up to 10x improvement in performance and 5x improvement in area compared to the same design described in generic VHDL.

The ideal core must be flexible, yet with predictable high-performance

To satisfy designers' need for shorter development time and high-performance, the ideal core must be flexible and support a wide range of FPGA sizes, speeds and packages. It must be optimized to the FPGA architecture for maximum performance and minimum area. Furthermore, to make it easy to use, the ideal core must allow pre-determined placement and routing for best predictability, and allow easy parameterization.

As seen in the comparison between soft cores and embedded hard cores, the ideal core is different for different applications. Hard cores are well suited for big, commonly used functions such as RAM, and extremely high-performance functions that cannot be implemented in standard FPGAs. The vast majority of core functions, however, will be provided as soft cores, as it maintains the flexibility, time-to-market and large-scale advantages of programmable logic. Thanks to rapid adoption of new process technologies and smart implementation techniques, soft cores provide superior flexibility, still with cost and performance in par with embedded hard cores.

Author:	Per Holmberg LogiCORE Product Manager CORE Solutions Group
Company:	Xilinx, Inc 2100 Logic Drive San Jose, CA 95124-3450
Phone:	(408) 879-5318
Fax:	(408) 377-3259
Email:	per@xilinx.com
Biography:	Per Holmberg has been active in the FPGA market for 10 years in sales management as well as product marketing positions. With Xilinx Per is the product manager for LogiCORE Solutions, a leading program for delivering core IP to the FPGA market. Prior experience includes ASIC design support and EDA tool integration and training at Ericsson Telecom in Sweden.