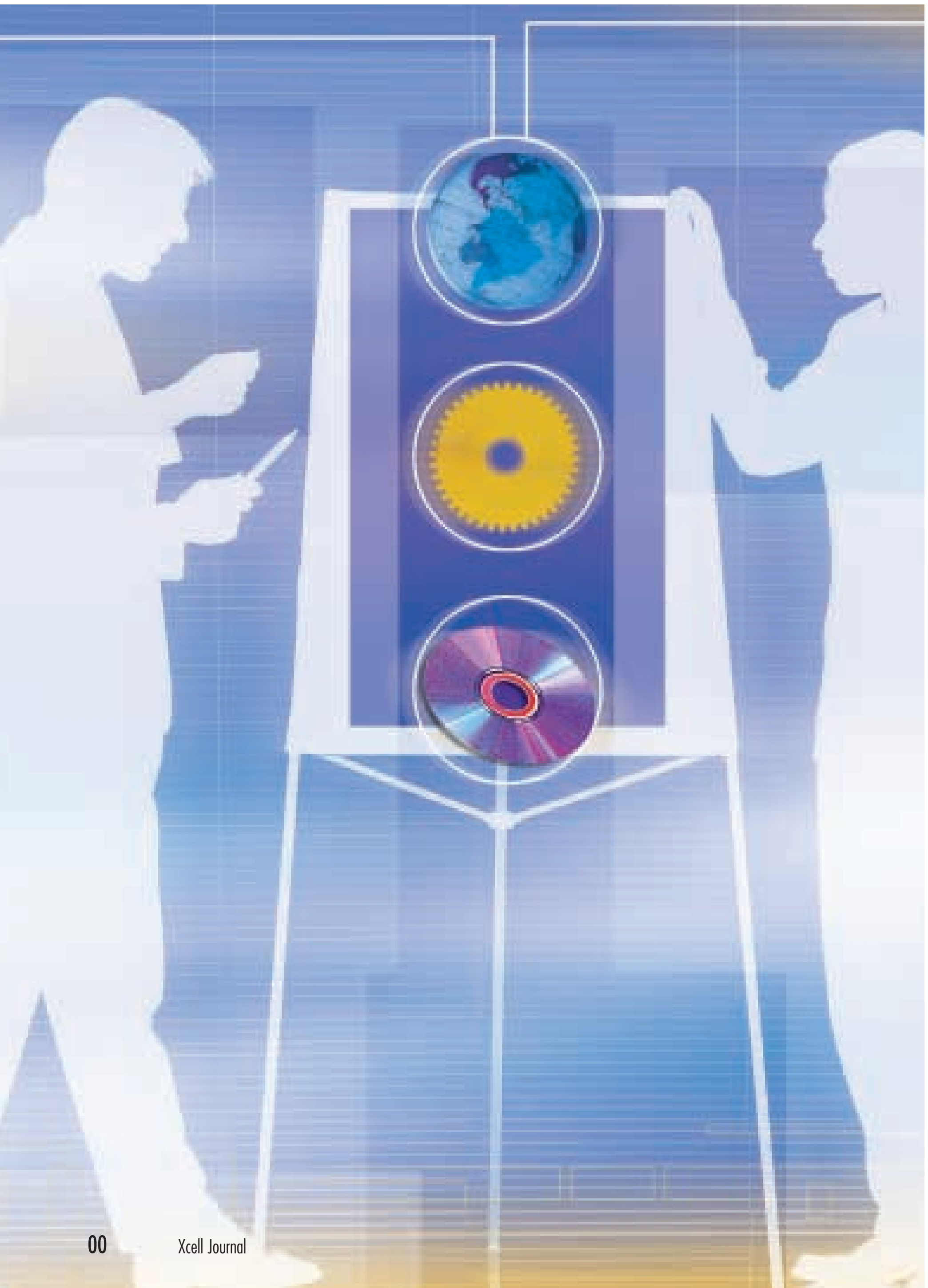
Exploring Hardware/Software Co-Design with Virtex-II Pro FPGAs

To explore the possibilities of hardware/software co-design, the Xilinx Design Services team created a reference design for a real-time operating system (RTOS) for telecommunications – here is an interview with the team.



by Jean-Louis Brelet, Manager
APD Technical Market Research
Xilinx, Inc.
jean-louis.brelet@xilinx.com

In this interview, Jean-Louis Brelet, manager of Technical Market Research for the Xilinx Advanced Products Division, interviewed Xilinx Design Services (XDS) software and hardware engineers about their practical experience with an RTOS reference design project. Senior Project Engineer Matthew Roche acted as spokesman for XDS.

What is hardware/software co-design?

Hardware/software co-design generally means that the hardware and software are developed at the same time – some software functions may be implemented in hardware for additional speed, and some hardware functions may be implemented in software to free up logic resources.

What does XDS do?

Xilinx Design Services, located in Dublin, Ireland, is part of the Xilinx Global Services Division (GSD). XDS has two functional groups: software engineers who provide custom software solutions, and hardware engineers who provide custom hardware solutions. The combined team consists of more than 50 engineers.

Typically, we help customers with their full product life cycle, from requirements capture, through design implementation and test, to final acceptance and delivery. We work very closely with our customers, and form a team of experts who best match the job requirements.

What are your previous experiences with embedded software?

We have worked on technologies that include:

- Digital video systems – both set-top boxes and head-ends
- High speed digital network communications systems
- Automotive applications
- Real time operating systems
- Board-level designs.

Due to the nature of design services, we have attained experience from many different development environments and methodologies. We have experience with a wide range of designs, for device-level to system-level applications.

How would you describe the Virtex-II Pro FPGA family from a software perspective?

The introduction of PowerPC™ processors into the Virtex-II Pro™ Platfrom FPGA architecture creates many new possibilities. Now we can design custom co-processors with innovative (and fast) interfaces. Due to the flexibility of the embedded IBM processors, and their surrounding programmable logic, we can offload some parts of our software design into a dedicated hardware implementation that can run much faster.

Previous hardware/software designs have been based around an inflexible hardware/software interface.

Jean-Louis Brelet

When did you start to design for the next-generation Virtex-II Pro Platform FPGA?

As soon as the technology became available, it became apparent that we must evaluate the system in a real project situation – the RTOS reference design. This way we could become aware of the new possibilities offered by the Virtex-II Pro device – and be best placed to contribute our knowledge of the advanced architectures used by our customers.

Project Objectives

What are the main objectives of this project?

The overall and real objective was to gain experience in developing a product in a tightly coupled hardware/software environment. The ability to shift functions from software to hardware, or vice versa, in a very short time frame, creates an environment where the design teams need a much greater knowledge of each other's work.

The smaller and more restricted objective was to identify the speed bottlenecks in the RTOS and remove the bottlenecks through the use of dedicated hardware – effectively implementing large sections of the kernel in hardware.

Before Virtex-II Pro FPGAs, how would you develop a system like this?

The RTOS project is very suited to the Virtex-II-Pro architecture. Given the costs, project time, cycle time, and risk required to develop this project with other technologies – for example, an ASIC – it is unlikely that it would even be undertaken. An ASIC design doesn't offer you the benefit of reprogrammability and flexibility to change/improve or upgrade the system in the field.

What is the expertise required to successfully implement these systems?

Software people must understand the nature of hardware designs and the type of problems encountered by the hardware team. They also must understand the possibilities and capabilities of the hardware.

Mathew Roche

Likewise, the hardware team must have a good understanding of software and how the application operates. Both teams must have a common language – or a good understanding of each other's language and a willingness to adapt.

Partitioning

What is your approach at the system level to partition embedded hardware and software functions?

We designed and implemented the RTOS reference design project for telecommunications with resource-constrained processors. We built it for state-based SDL (specification and design language) applications.

The application for this RTOS was to be in systems with high numbers of processes with large numbers of context switches. At this level, the overhead of the operating system would be significant and thus, must be minimized.

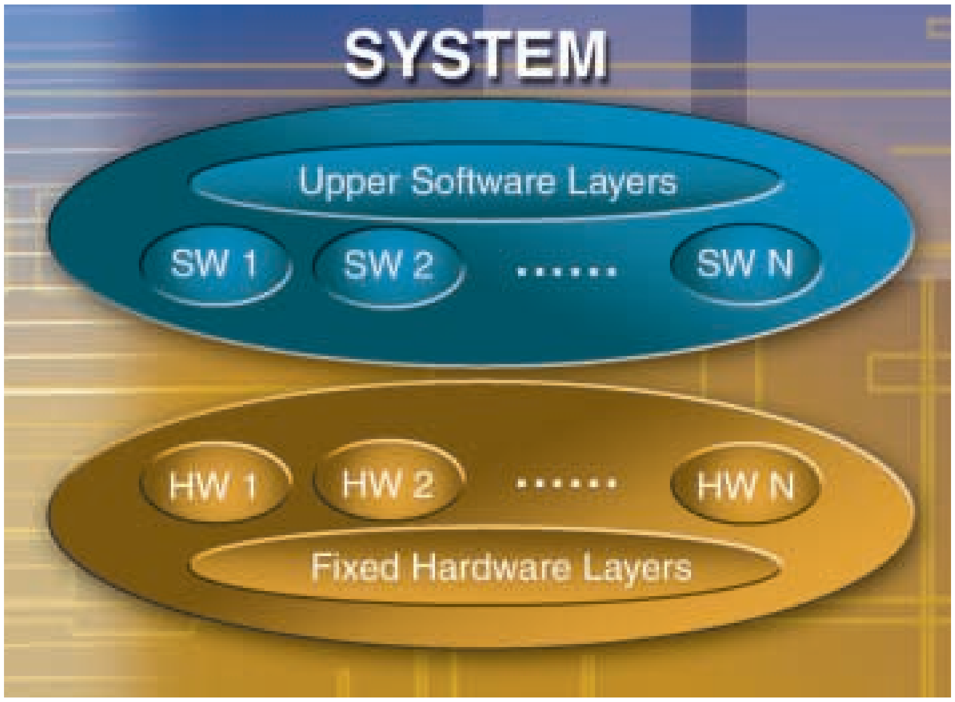
First, we analyzed the system and identified the time-hungry elements. Our initial performance analysis indicated that processing of the state tables and managing the most heavily used queues (signal queues) were the processor-intensive tasks. The first iterations of the design placed these tasks into hardware with a clearly defined hardware/software interface. This did not completely remove the bottlenecks, but it did have a tendency to move them from the processing into the interface.

Further iterations of the design required much closer integration of the hardware and software teams. This allowed the design of coherent and integrated tasks, which could be completely dealt with in hardware, and required minimal use of the interface.

Having the processor embedded into the FPGA fabric allows a variety of interface techniques, which do not necessarily require the use of the IBM CoreConnect™ processor local bus. This feature became an important design issue, as the speed of the system was capable of hogging the bus.

In our experience, we have found that there is no distinct division between functions that are suitable for hardware or software. It becomes a trade-off of how fast you want to do something and how many configurable logic blocks you want to spend on doing it. Greater complexity of the function or required speed of operation will bias the design towards software or hardware.

At the system design level, the division of labor became less important as the project went on. It was necessary for both sides to have a good understanding of what the other team was doing. Design decisions had to be made with knowledge of both hardware and software. This had some interesting implications for the language and terminology used by the teams.



How would you summarize your method for partitioning the system?

We simply integrated the design teams, to avoid a purely hardware- or software-led solution.

How have you split the teamwork?

Most of the software was designed first, because we needed to understand how the system would operate before transferring parts of it into hardware. This gave us the advantage of having functioning prototypes before a full hardware implementation – significantly reducing our theoretical time to market.

Virtex-II Pro Embedded Software

Is it easier to create embedded software for the Virtex-II Pro fabric?

In the first applications, embedded software development techniques will not necessarily be different for Virtex-II Pro devices – they can be viewed as conventional systems with performance improvements and reduced chip count. However, as experience builds, it will become apparent that traditional design methods do not use the full potential of the device. System partitioning will become part of the architectural design process, and the boundary between hardware and software will become less rigid.

The RTOS project gave us an opportunity to examine the project with system co-design as the goal. Virtex-II-Pro Platform FPGAs allow significantly more integrated designs with all the performance benefits that entails.

What is the most innovative feature of Virtex-II Pro for a software engineer?

Using multiple processors (both IBM PowerPC 405 hard core and Xilinx MicroBlaze™ soft core processors), on a single chip, allow you to dedicate processes to particular processors for critical tasks. You can easily implement interfaces between the different processors, using hardware-based FIFOs, reducing the communications bus overhead.

How easy is it to partition a Virtex-II Pro design?

Partitions are very flexible in the Virtex architecture. Changes were being made to the interface all the way through the design and implementation of our RTOS system. This allowed us to reduce our design time and minimize design roadblocks.

Conclusion

Could you summarize your experience?

Close cooperation between hardware and software teams is critical.

Is this project an example of making the impossible possible?

No. This project does not do anything that has been impossible before. What it does do is make it faster, cheaper, and with reduced risk.

How can Xilinx customers benefit from your expertise?

It is part of our culture to maintain high levels of expertise in advanced and state-of-the-art technological systems and methodologies. As project managers, we are constantly improving our service and quality in dealing with our customers' projects. Increasing competitiveness in the marketplace means that there is a higher emphasis on time to market and quality. XDS continues to improve its skills in order to help our customers achieve these challenging targets.

XDS can be contacted by e-mail at *designservices@xilinx.com* or by visiting *http://support.xilinx.com/xds/*.